LDBC

The graph & RDF benchmark reference

The LDBC Financial Benchmark (version 0.0.1-SNAPSHOT)

The specification was built on the source code available at

https://github.com/ldbc/ldbc_finbench_docs

Abstract

TODO

Executive Summary

TODO

Table of Contents

Acknowledgments

Special thanks to all the people that have contributed to the development of this benchmark:

- Shipeng Qi (AntGroup)
- Heng Lin (AntGroup)
- Chuntao Hong (AntGroup)
- Xiaowei Zhu (AntGroup)

Definitions

Datagen: Is the data generator provided by the LDBC FinBench, which is responsible for generating the data needed to run the benchmark.

DBMS: A DataBase Management System.

LDBC FinBench: Linked Data Benchmark Council Social Network Benchmark.

Query Mix: Refers to the ratio between read and update queries of a workload, and the frequency at which they are issued.

SF (Scale Factor): The LDBC FinBench is designed to target systems of different size and scale. The scale factor determines the size of the data used to run the benchmark, measured in Gigabytes.

SUT: The System Under Test is defined to be the database system where the benchmark is executed.

Test Driver: A program provided by the LDBC FinBench, which is responsible for executing the different workloads and gathering the results.

Full Disclosure Report (FDR): The FDR is a document which allows reproduction of any benchmark result by a third party. This contains complete description of the SUT and the circumstances of the benchmark run, e.g. configuration of SUT, dataset and test driver, etc.

Test Sponsor: The Test Sponsor is the company officially submitting the Result with the FDR and will be charged the filing fee. Although multiple companies may sponsor a Result together, for the purposes of the LDBC processes the Test Sponsor must be a single company. A Test Sponsor need not be a LDBC member. The Test Sponsor is responsible for maintaining the FDR with any necessary updates or corrections. The Test Sponsor is also the name used to identify the Result.

Workload: A workload refers to a set of queries of a given nature (i.e. interactive, analytical, business), how they are issued and at which rate.

1 Introduction

1.1 Practice basis for FinBench

Based on a comprehensive survey on financial scenarios inside and outside AntGroup, we summarize some common user cases in risk control, AML (Anti-Money Laundering), KYC(Know Your Customer), and so on. With the best practices in the industry, we propose this design for LDBC Financial Benchmark.

1.2 Design concepts of FinBench

1.2.1 Data Schema

The data design is based on a review of actual data in systems. The data schema for FinBench is inspired by the real data in financial systems. Stored data in systems contain entities in the real world including accounts, medium(device, IP, etc.), persons, companies, orders, loans, and so on. These entities are Nodes in the data schema. And the edges in the data schema reflect financial actions in the real world like transferring funds from one account to another, guaranteeing by one person for another. The designed data schema is specified in Section 2.3.

1.2.2 Load definition

In this draft, we do not finish the load design with the workloads. But we conclude some patterns of load after reviewing audit logs of systems. They are:

- Data read and write intermittently with random intervals.
- There are some light and extremely heavy loads periodically.
- Large scale data ETLs are triggered when at midnight or some time the system load is light.
- Tight latency constraints.

1.2.3 Workloads

Compared with LDBC SNB, we divide the queries into three workloads based on their complexity and application latency. They are:

- Online Workload (See Section 4). This workload is supposed to include cases in online applications which are expected to be finished in low latency. The latency may range from tens to hundreds in millisecond precision. The cases are usually accessing at most 3 step neighborhood from a start node. In this draft, we fill cases in the online workload for discussion.
- Nearline Workload (See Section 5). This workload is supposed to include cases in nearline applications which are expected to be finished in higher latency than online applications but lower latency than offline applications. The latency may range from several seconds to several minutes. The cases usually include subgraph traversal, pattern matching, and deeper neighborhood accessing. Nearline workload will be designed in the future.
- Offline Workload (See Section 6). This workload is supposed to include cases in offline applications which are expected to be finished in high latency. The latency may range from tens of minutes and even many hours. The cases are usually performing iterative graph analytics. Offline workload will be designed in the future.

In Online workload, the queries include read queries, write queries, and read-write queries. Read-write query is a significant design that reflects the complexity of financial systems. In real-time risk control, risk analysis for involved accounts is supposed to be triggered intermediately when some deals related are recorded. Abstracting from such scenarios, we propose a concept Read-write Query. A read-write query is composed of read queries and write queries consecutively. For example, a read-write query is composed of inserting transfer edges(a write query), risk analysis for a specific account(a read query), in the example mentioned above. For detail, see Section 4.3.

1.3 Differences between FinBench and SNB

We highlight several differences between FinBench and SNB as the following:

- Multiple edges can exist between two vertices, e.g., many money-transfers can occur between two accounts each day.
- Read-write queries, which is a query sequence with a mix of read and write queries.
- Filtering with backward dependency in variable-length paths, e.g., finding all money-transfer paths A ->[e1]->B-[e2]->...->X in which the timestamp of each transfer ei is larger than that of ei-1
- Property update queries, e.g., marking an account as blocked or high-risk for risk control.
- Latency sensitive, e.g., some queries need to return in less than 10ms.

2 Benchmark Specification

2.1 Requirements

[TODO. This section will be filled after benchmark software is designed and developed.]

2.2 Software and Useful Links

[TODO. This section will be filled after benchmark software is designed and developed.]

2.3 Data

2.3.1 Data Types

Table 2.1 describes the different data types used in the benchmark.

Type	Description
ID	integer type with 64-bit precision. All IDs within a single entity type (e.g. Person) are unique, but different entity types (e.g. a Person and an Account) might have the same ID.
32-bit Integer	integer type with 32-bit precision
64-bit Integer	integer type with 64-bit precision
String	variable length text of size 40 Unicode characters
Long String	variable length text of size 256 Unicode characters
Text	variable length text of size 2000 Unicode characters
Date	date with a precision of a day, encoded as a string with the following format: *yyyy-mm-dd*, where *yyyy* is a four-digit integer representing the year, the year, *mm* is a two-digit integer representing the month and *dd* is a two-digit integer representing the day.
DateTime	date with a precision of milliseconds, encoded as a string with the following format: *yyyy-mm-ddTHH:MM:ss.sss+0000*, where *yyyy* is a four-digit integer representing the year, the year, *mm* is a two-digit integer representing the month and *dd* is a two-digit integer representing the day, *HH* is a two-digit integer representing the hour, *MM* is a two digit integer representing the minute and *ss.sss* is a five digit fixed point real number representing the seconds up to millisecond precision. Finally, the *+0000* of the end represents the timezone, which in this case is always GMT.
Boolean	logical type, taking the value of either True of False

Table 2.1: Description of the data types.

2.3.2 Data Schema

Figure 2.1 shows the data schema in UML. The schema defines the structure of the data used in the benchmark in terms of entities and their relations. Data represents a snapshot of the activity in several financial scenarios during a period of time. The schema specifies different entities, their attributes, and their relations. All of them are described in the following sections.

Note: The dashed arrows in the schema represent multiple edges which means there are more than one edge from start node to end node.

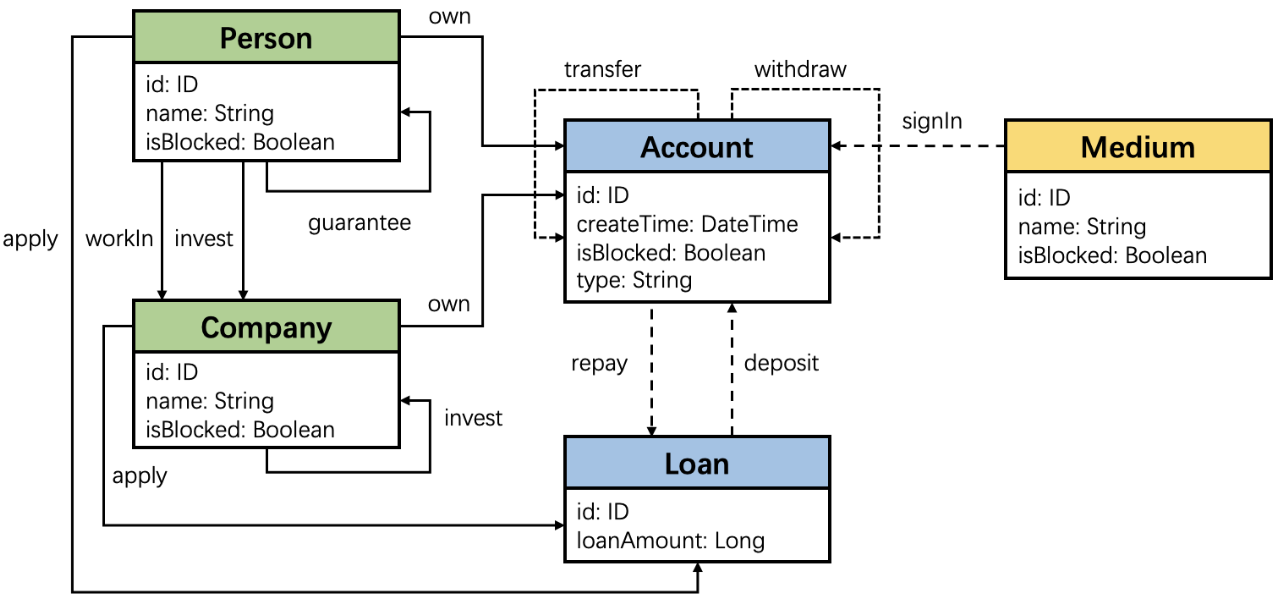


Figure 2.1: The LDBC FinBench data schema

2.3.2.1 Entities

Person: a person of the real world. Table 2.2 shows the attributes.

Attribute	Type	Description
`id`	ID	The identifier of the person.
`name`	String	The name of the person.
`isBlocked`	Boolean	If the person is blocked or concerned in systems.

Table 2.2: Attributes of Person entity.

Company: a company of the real world, which persons work in and persons invest. Table 2.3 shows the attributes.

Attribute	Type	Description
`id`	ID	The identifier of the company.
`name`	String	The name of the company.
`isBlocked`	Boolean	If the company is blocked or concerned in systems.

Table 2.3: Attributes of Company entity.

Account: an account in real world financial systems, which is registered and owned by persons and companies. It includes many types such as personalDeposit, personalCredit, etc. It can deal with other accounts. Table 2.4 shows the attributes.

Attribute	Type	Description
`id`	ID	The identifier of the account.
`createTime`	DateTime	The time when the account created.
`isBlocked`	Boolean	If the account is blocked or concerned in systems.
`Type`	String	The type of Account including personalDeposit, personalCredit, companyDeposit, card.

Table 2.4: Attributes of Company entity.

Loan: a loan for persons and company to apply in real world. Table 2.5 shows the attributes.

Attribute	Type	Description
id	ID	The identifier of the loan.
loanAmount	64-bit Integer	the amount of a loan

Table 2.5: Attributes of Company entity.

Medium: an abstract standing for things that users use to sign in account in real world, such as IP, mac, phone numbers. Table 2.6 shows the attributes.

Attribute	Type	Description
id	ID	The identifier of the medium.
name	String	The name of the medium.
isBlocked	Boolean	If the medium is blocked or concerned in systems.

Table 2.6: Attributes of Medium entity.

2.3.2.2 Relations

Relations connect entities of different types.

Name	Tail	Head	Multiplicity	Description
signIn	Medium	Account	N	An account is signed in with a Media 1.timestamp: DateTime
own	Person/Company	Account	1	A person or a company owns an account.
transfer	Account	Account	N	Fund transfers between two accounts. 1.timestamp: DateTime 2.amount: 64-bit Integer
deposit	Loan	Account	N	Loan fund is deposited to an account 1. timestamp: DateTime 2. amount: 64-bit Integer
repay	Account	Loan	N	Loan is repaid from an account 1. timestamp: DateTime 2. amount: 64-bit Integer
withdraw	Account	Account	N	Fund is transferred from an account to another account whose type is card 1. timestamp: DateTime 2. amount: 64-bit Integer
invest	Person/Company	Company	1	A person or a company invests a company 1. timestamp: DateTime 2. percent: Float
workIn	Person	Company	1	A person works in a company /
apply	Person/Company	Loan	1	A person or a company applies a Loan. 1. timestamp: DateTime
guarantee	Person	Person	1	A person guarantees another for some reason like loans. 1. timestamp: DateTime

Table 2.7: Description of the data relations.

2.3.3 Data Generation

[TODO. This section will be filled after benchmark software designed and developed.]

2.3.4 Output Data

[TODO. This section will be filled after benchmark software designed and developed.]

2.4 Benchmark Workflow

[TODO. This section will be filled after benchmark software designed and developed.]

3 Workloads

3.1 Query Description Format

[TODO. This section will be filled further after draft is approved.]

3.2 Substitution Parameters

[TODO. This section will be filled further after draft is approved.]

3.3 Load Definition

[TODO. This section will be designed further after draft is approved. The design concepts are listed in Section 1.2.2]

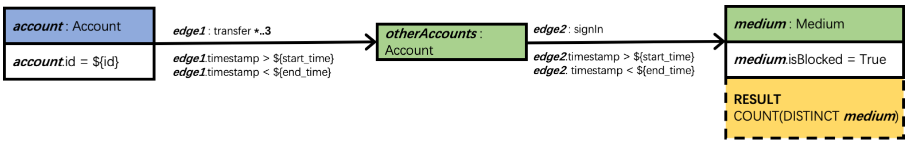
4 Online Workload

This workload consists of a set of relatively simple read queries, write queries and read-write operations, that touch a significant amount of data. These queries and operations are usually considered as online data processing and analysis in online financial systems. The LDBC FinBench Online workload consists of three query classes:

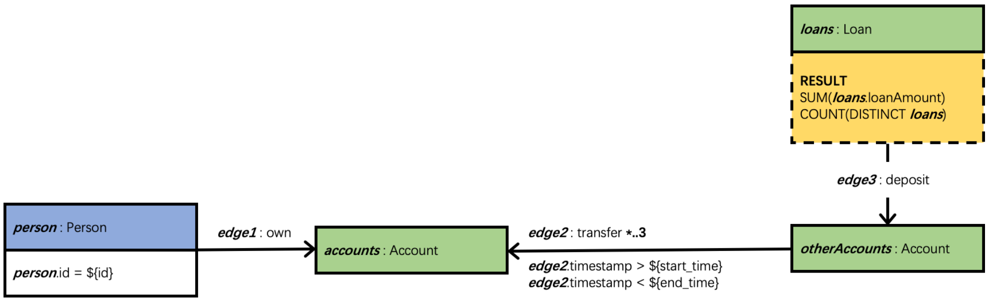
- Read queries. See Section 4.1. This section contains many basic read queries that are typical in financial scenarios.
- Write queries. See Section 4.2. This section contains many basic write queries that are typical in financial scenarios.
- Read-write queries. See Section 4.3. This section contains many read-write operations composed of basic reads and writes in section 4.1 and 4.2. In each operation, the basic reads and writes are supposed to launch step by step consecutively. The result may be wrong if the previous reads and writes are not be processed accurately in time. This feature of FinBench is a big difference from LDBC SNB.

4.1 Read Queries

Online / read / 1

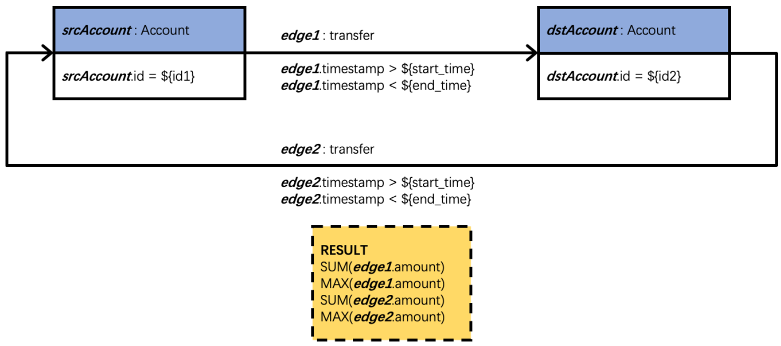
query	Online / read / 1
title	Blocked medium related accounts
pattern	
desc.	Given a start Account, find Accounts which is connected to a blocked Medium via the signIn relationship by at most 3 steps via the transfer relationship. Note that the timestamps of all the relationships are in a specific time range between start_time and end_time. Note that the transfer path of edge1(at most 3 steps transfer relationship) are in ascending order in terms of timestamp. Return the count of distinct medium.
params	1 | `id` | ID | id of the start Account 2 | `start_time` | DateTime | begin of the time window 3 | `end_time` | DateTime | end of the time window
result	1 | `COUNT(DISTINCT meidum)` | 64-bit Integer | R | number of different mediums
sort	1 | todo | ↑
limit	todo
CPs	0.0
relevance	This query is a typical fund transfer cycle analysis case in AWL.

Online / read / 2

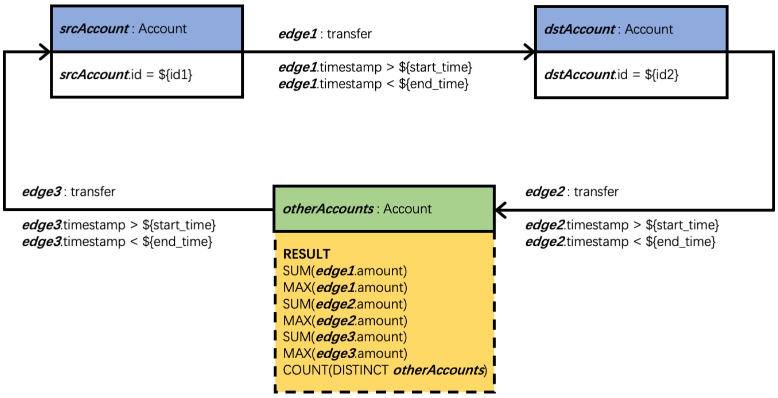
query	Online / read / 2
title	example
pattern	
desc.	Given a Person, find an Account owned by the Person which has funds transferred from other Accounts by at most 3 steps which connected to loan via deposit relationship in a specific time range between start_time and end_time. Note that the transfer path of edge2(at most 3 steps transfer relationship) are in ascending order in terms of timestamp. Return the sum and count of loans fund amount.
params	1 | id | ID | id of the start Person 2 | start_time | DateTime | begin of the time window 3 | end_time | DateTime | end of the time window
result	1 | SUM(loans.loanAmount) | 64-bit Integer | R | sum of all loans 2 | COUNT(DISTINCT loans) | 64-bit Integer | R | number of different loans
sort	1 | todo | ↑
limit	todo
CPs	0.0
relevance	This query is typical gang analysis case in risk control.

Online / read / 3

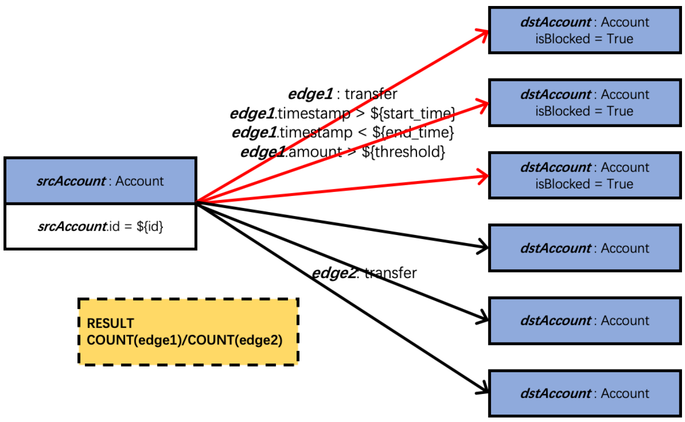
OR 1
OR 2
OR 3
OR 4
OR 5
OR 6
OR 7
OR 8
OR 9
OR 10
OR 11
OR 12
OR 13
OR 14
OR 15
OR 16
OR 17
OR 18
OR 19
OR 20
OR 21
OR 22
OR 23
OR 24

<table>
<tr><td>query</td><td>Online / read / 3</td></tr>
<tr><td>title</td><td>Anti-Money Laundering: Two Accounts in a cycle</td></tr>
<tr><td>pattern</td><td>
</td></tr>
<tr><td>desc.</td><td>Given two Accounts, find the sum and max of fund transfer between them in a specific time range between start_time and end_time. Note that the timestamp of edge2 is bigger than the one of edge1. Return the sum and max of amount of edge1 and edge2.</td></tr>
<tr><td>params</td><td>
<table>
<tr><td>1</td><td>id1</td><td>ID</td><td>id of the src Account</td></tr>
<tr><td>2</td><td>id2</td><td>ID</td><td>id of the dst Account</td></tr>
<tr><td>3</td><td>start_time</td><td>DateTime</td><td>begin of the time window</td></tr>
<tr><td>4</td><td>end_time</td><td>DateTime</td><td>end of the time window</td></tr>
</table></td></tr>
<tr><td>result</td><td>
<table>
<tr><td>1</td><td>SUM(edge1.loanAmount)</td><td>64-bit Integer</td><td>R</td><td>sum of transfers from srcAccount to dstAccount</td></tr>
<tr><td>2</td><td>MAX(edge1.loanAmount)</td><td>64-bit Integer</td><td>R</td><td>max of transfers from srcAccount to dstAccount</td></tr>
<tr><td>3</td><td>SUM(edge2.loanAmount)</td><td>64-bit Integer</td><td>R</td><td>sum of transfers from dstAccount to srcAccount</td></tr>
<tr><td>4</td><td>MAX(edge2.loanAmount)</td><td>64-bit Integer</td><td>R</td><td>max of transfers from dstAccount to srcAccount</td></tr>
</table></td></tr>
<tr><td>sort</td><td>
<table>
<tr><td>1</td><td>todo</td><td>↑</td><td></td></tr>
</table></td></tr>
<tr><td>limit</td><td>todo</td></tr>
<tr><td>CPs</td><td>0.0</td></tr>
<tr><td>relevance</td><td>This query is a typical fund transfer cycle analysis case in AWL.</td></tr>
</table>

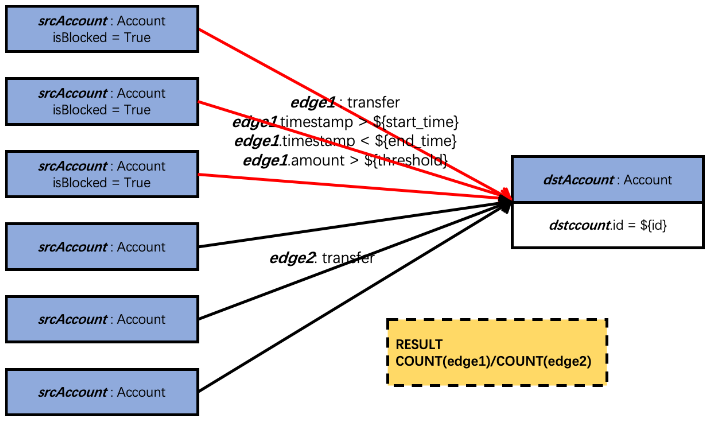
Online / read / 4

<table>
<tr><td>query</td><td>Online / read / 4</td></tr>
<tr><td>title</td><td>Anti-Money Laundering: Three Accounts in a cycle</td></tr>
<tr><td>pattern</td><td>
</td></tr>
<tr><td>desc.</td><td>Given two Accounts, find the sum and max of transfer of them via a third-party Account in a specific time range between start_time and end_time. Note that the transfer of edge1, edge2, edge3 are in ascending order in terms of timestamp. Return the sum and max of amount of edge1, edge2, edge3 and the count of distinct of third-party account.</td></tr>
<tr><td>params</td><td><table>
<tr><td>1</td><td>id1</td><td>ID</td><td>id of the src Account</td></tr>
<tr><td>2</td><td>id2</td><td>ID</td><td>id of the dst Account</td></tr>
<tr><td>3</td><td>start_time</td><td>DateTime</td><td>begin of the time window</td></tr>
<tr><td>4</td><td>end_time</td><td>DateTime</td><td>end of the time window</td></tr>
</table></td></tr>
<tr><td>result</td><td><table>
<tr><td>1</td><td>SUM(edge1.loanAmount)</td><td>64-bit Integer</td><td>R</td><td>sum of transfers from srcAccount to dstAccount</td></tr>
<tr><td>2</td><td>MAX(edge1.loanAmount)</td><td>64-bit Integer</td><td>R</td><td>max of transfers from srcAccount to dstAccount</td></tr>
<tr><td>3</td><td>SUM(edge2.loanAmount)</td><td>64-bit Integer</td><td>R</td><td>sum of transfers from dstAccount to otherAccount</td></tr>
<tr><td>4</td><td>MAX(edge2.loanAmount)</td><td>64-bit Integer</td><td>R</td><td>max of transfers from dstAccount to otherAccount</td></tr>
<tr><td>5</td><td>SUM(edge3.loanAmount)</td><td>64-bit Integer</td><td>R</td><td>sum of transfers from otherAccount to srcAccount</td></tr>
<tr><td>6</td><td>MAX(edge3.loanAmount)</td><td>64-bit Integer</td><td>R</td><td>max of transfers from otherAccount to srcAccount</td></tr>
</table></td></tr>
<tr><td>sort</td><td><table><tr><td>1</td><td>todo</td><td>↑</td><td></td></tr></table></td></tr>
<tr><td>limit</td><td>todo</td></tr>
<tr><td>CPs</td><td>0.0</td></tr>
<tr><td>relevance</td><td>This query is a typical fund transfer cycle analysis case in AWL.</td></tr>
</table>

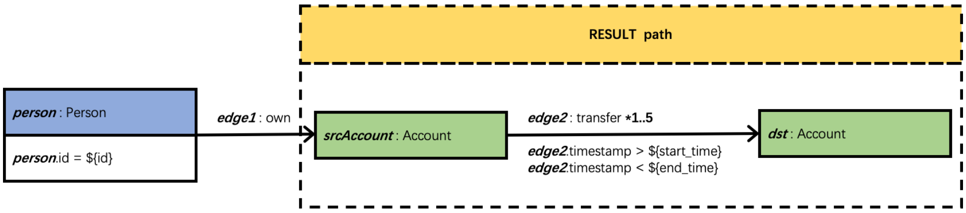
Online / read / 5

<table>
<tr><td>query</td><td>Online / read / 5</td></tr>
<tr><td>title</td><td>One-to-many blocked account monitoring</td></tr>
<tr><td>pattern</td><td>
</td></tr>
<tr><td>desc.</td><td>Given an Account, find the ratio of transfer-outs to blocked Accounts in all its transfer-outs in a specific time range between start_time and end_time. Return the ratio.</td></tr>
<tr><td>params</td><td><table>
<tr><td>1</td><td>id</td><td>ID</td><td>id of the srcAccount</td></tr>
<tr><td>2</td><td>threshold</td><td>64-bit Integer</td><td>threshold of transfer amount</td></tr>
<tr><td>3</td><td>start_time</td><td>DateTime</td><td>begin of the time window</td></tr>
<tr><td>4</td><td>end_time</td><td>DateTime</td><td>end of the time window</td></tr>
</table></td></tr>
<tr><td>result</td><td><table><tr><td>1</td><td>blockRatio</td><td>float</td><td>R</td><td></td></tr></table></td></tr>
<tr><td>sort</td><td><table><tr><td>1</td><td>todo</td><td>↑</td><td></td></tr></table></td></tr>
<tr><td>limit</td><td>todo</td></tr>
<tr><td>CPs</td><td>0.0</td></tr>
<tr><td>relevance</td><td>This query is a typical risky account recognition case in risk control.</td></tr>
</table>

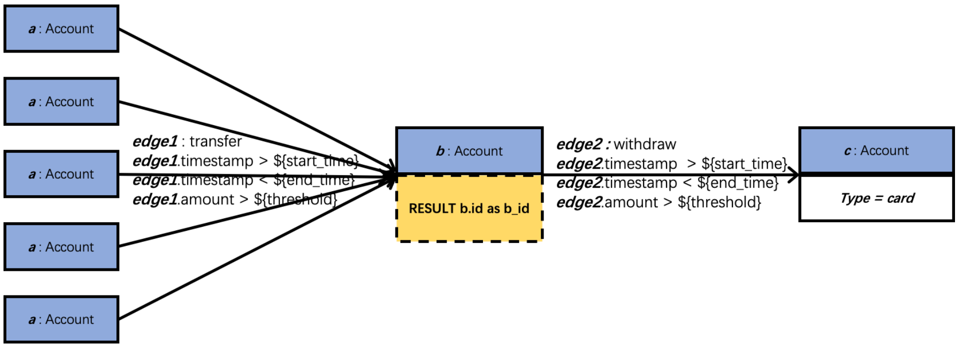
Online / read / 6

query	Online / read / 6
title	Many-to-one blocked account monitoring
pattern	
desc.	Given an Account, find the ratio of transfer-ins from blocked Accounts in all its transfer-ins in a specific time range between start_time and end_time. Return the ratio.
params	1 | id | ID | id of the dstAccount 2 | threshold | 64-bit Integer | threshold of transfer amount 3 | start_time | DateTime | begin of the time window 4 | end_time | DateTime | end of the time window
result	1 | blockRatio | float | R
sort	1 | todo | ↑
limit	todo
CPs	0.0
relevance	This query is a typical risky account recognition case in risk control.

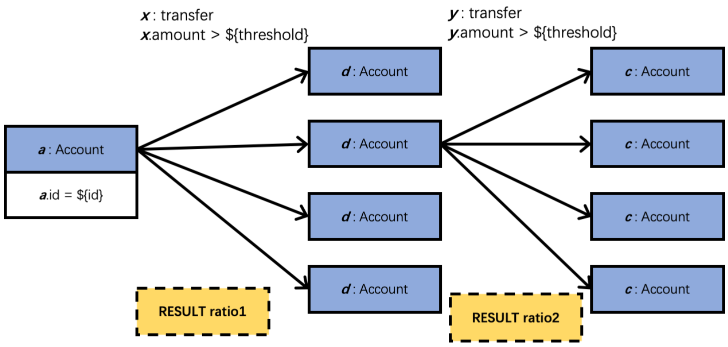
Online / read / 7

<table>
<tr><td>query</td><td>Online / read / 7</td></tr>
<tr><td>title</td><td>Exact Account Transfer Trace</td></tr>
<tr><td>pattern</td><td>
</td></tr>
<tr><td>desc.</td><td>Given a Person, find the paths from the Account the Person owned to other Accounts by at most 5 steps via transfer relationship in a specific time range between start_time and end_time. Note that the transfer path of edge2 are in ascending order in terms of timestamp. Return the paths.</td></tr>
<tr><td>params</td><td><table><tr><td>1</td><td>id</td><td>ID</td><td>id of the start Person</td></tr><tr><td>2</td><td>start_time</td><td>DateTime</td><td>begin of the time window</td></tr><tr><td>3</td><td>end_time</td><td>DateTime</td><td>end of the time window</td></tr></table></td></tr>
<tr><td>result</td><td><table><tr><td>1</td><td>path</td><td>?</td><td>R</td><td></td></tr></table></td></tr>
<tr><td>sort</td><td><table><tr><td>1</td><td>todo</td><td>↑</td><td></td></tr></table></td></tr>
<tr><td>limit</td><td>todo</td></tr>
<tr><td>CPs</td><td>0.0</td></tr>
<tr><td>relevance</td><td>This query is a typical fund transfer trace case in risk control.</td></tr>
</table>

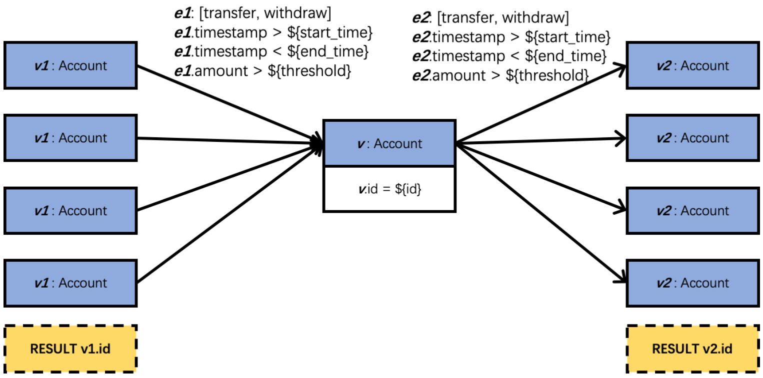
Online / read / 8

<table>
<tr><td>query</td><td>Online / read / 8</td></tr>
<tr><td>title</td><td>Withdrawal after Many-to-One transfer</td></tr>
<tr><td>pattern</td><td>
</td></tr>
<tr><td>desc.</td><td>Find all the accounts that match the requirements below:
More than 5 transfer-ins from other Accounts the Account whose amount exceeds threshold in a specific time range between start_time and end_time.
The amount of withdrawal to another account exceeds threshold after transfer-ins in a specific time range between start_time and end_time.
Transfer path of edge1, edge2 are in ascending order in terms of timestamp.
Return all the accounts' id.</td></tr>
<tr><td>params</td><td><table><tr><td>1</td><td>threshold</td><td>64-bit Integer</td><td>threshold of transfer amount</td></tr><tr><td>2</td><td>start_time</td><td>DateTime</td><td>begin of the time window</td></tr><tr><td>3</td><td>end_time</td><td>DateTime</td><td>end of the time window</td></tr></table></td></tr>
<tr><td>result</td><td><table><tr><td>1</td><td>b_id</td><td>ID</td><td>R</td><td>id of many-to-one Account</td></tr></table></td></tr>
<tr><td>sort</td><td><table><tr><td>1</td><td>todo</td><td>↑</td><td></td></tr></table></td></tr>
<tr><td>limit</td><td>todo</td></tr>
<tr><td>CPs</td><td>0.0</td></tr>
<tr><td>relevance</td><td>This query is a typical fund withdrawal after pooling funds case in risk control.</td></tr>
</table>

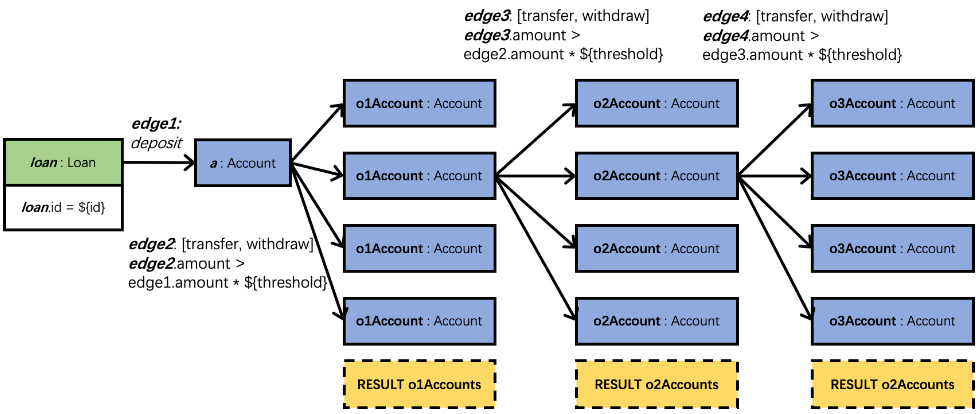
Online / read / 9

<table>
<tr><td>query</td><td>Online / read / 9</td></tr>
<tr><td>title</td><td>Exact account main transfer trace</td></tr>
<tr><td>pattern</td><td>
</td></tr>
<tr><td>desc.</td><td>Given an Account, find the ratio of each transfer-out at 1 step and at 2 steps. Note that the transfer path of x, y are in ascending order in terms of timestamp and in descending order in terms of amount. Return the ratios in descending order.</td></tr>
<tr><td>params</td><td><table><tr><td>1</td><td>id</td><td>ID</td><td>id of the start Account</td></tr><tr><td>2</td><td>threshold</td><td>64-bit Integer</td><td>threshold of transfer amount</td></tr></table></td></tr>
<tr><td>result</td><td><table><tr><td>1</td><td>ratio1</td><td>float</td><td>R</td><td></td></tr><tr><td>2</td><td>ratio2</td><td>float</td><td>R</td><td></td></tr></table></td></tr>
<tr><td>sort</td><td><table><tr><td>1</td><td>todo</td><td>↑</td><td></td></tr></table></td></tr>
<tr><td>limit</td><td>todo</td></tr>
<tr><td>CPs</td><td>0.0</td></tr>
<tr><td>relevance</td><td>This query is a typical main stream of fund transfer-out trace case in risk control.</td></tr>
</table>

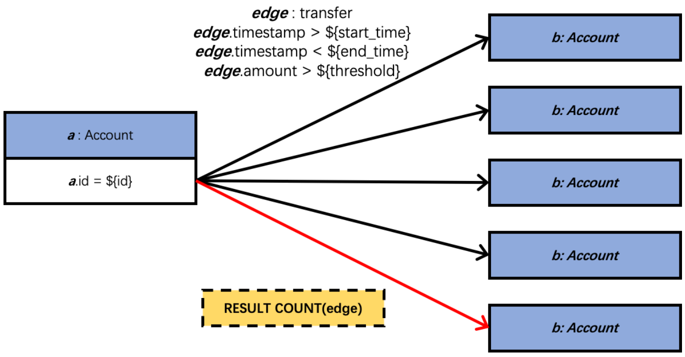
Online / read / 10

<table>
<tr><td>query</td><td>Online / read / 10</td></tr>
<tr><td>title</td><td>Fast-in and Fast-out</td></tr>
<tr><td>pattern</td><td>
</td></tr>
<tr><td>desc.</td><td>Given an Account, find all the transfer-in accounts and transfer-out accounts where the ratio of the sum of transfer-ins over the sum of transfer-outs is located in [0.8, 1.2] in a specific time range between start_time and end_time. Note that the transfer path of e1, e2 are in ascending order in terms of timestamp. Return the transfer-in accounts' ids and transfer-out accounts' ids.</td></tr>
<tr><td>params</td><td><table>
<tr><td>1</td><td><code>threshold</code></td><td>64-bit Integer</td><td>threshold of transfer amount</td></tr>
<tr><td>2</td><td><code>start_time</code></td><td>DateTime</td><td>begin of the time window</td></tr>
<tr><td>3</td><td><code>end_time</code></td><td>DateTime</td><td>end of the time window</td></tr>
</table></td></tr>
<tr><td>result</td><td><table>
<tr><td>1</td><td><code>v1.id</code></td><td>ID</td><td>R</td><td></td></tr>
<tr><td>2</td><td><code>v2.id</code></td><td>ID</td><td>R</td><td></td></tr>
</table></td></tr>
<tr><td>sort</td><td><table>
<tr><td>1</td><td><code>todo</code></td><td>↑</td><td></td></tr>
</table></td></tr>
<tr><td>limit</td><td>todo</td></tr>
<tr><td>CPs</td><td>0.0</td></tr>
<tr><td>relevance</td><td>This query is a typical fast-in and fast-out recognition case in risk control.</td></tr>
</table>

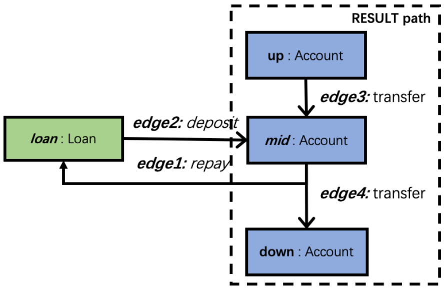
Online / read / 11

query	Online / read / 11
title	Transfer trace after loan applied
pattern	
desc.	Given a Loan, trace the main fund stream by 3 steps via transfer or withdraw relationship from the account the Loan connect to via deposit relationship. Return all the accounts at each step. Note that the transfer path of edge1, edge2, edge3, edge4 are in ascending order in terms of timestamp and in descending order in terms of amount. Return the sum of transfer/withdraw amount at each step.
params	1 | `id` | ID | id of the dstAccount 2 | `threshold` | 64-bit Integer | threshold of transfer amount
result	1 | `o1Account` | [ID] | R 2 | `o2Account` | [ID] | R 3 | `o3Account` | [ID] | R
sort	1 | `todo` | ↑
limit	todo
CPs	0.0
relevance	This query is a typical main fund trace case after loan granted in risk control.

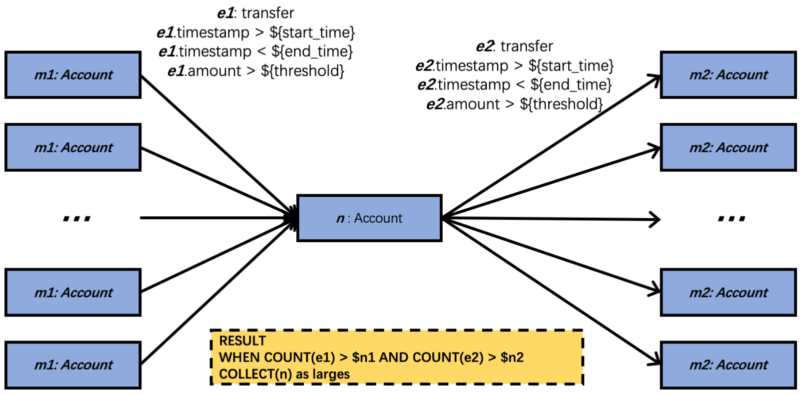
Online / read / 12

query	Online / read / 12
title	Abnormal transfer from an account
pattern	
desc.	Given an Account, find all the transfer-outs from the Account whose amount exceeds threshold in a specific time range between start_time and end_time. Return the count of transfer-outs.
params	1 | `id` | ID | id of the dstAccount 2 | `threshold` | 64-bit Integer | threshold of transfer amount
result	1 | `COUNT(edge)` | 32-bit Integer | R
sort	1 | todo | ↑
limit	todo
CPs	0.0
relevance	This query is a typical frequent transfer-out recognition case in risk control.

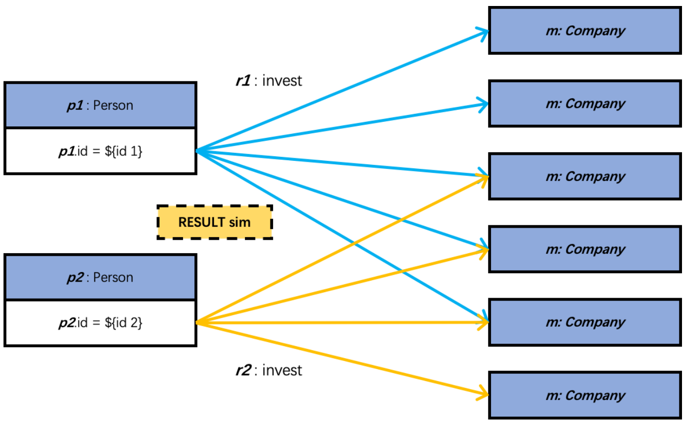
Online / read / 13

<table>
<tr><td>query</td><td>Online / read / 13</td></tr>
<tr><td>title</td><td>Money laundering via loans</td></tr>
<tr><td>pattern</td><td>
</td></tr>
<tr><td>desc.</td><td>Find all the accounts that match the requirements below: * Deposited from a Loan where the amount exceeds threshold * Repay to the Loan in a short period after loan deposit and the amount repaid exceeds a ratio of the deposit amount.
Note that the timestamps of transfer edges are in order: edge2 < edge4 < edge3 < edge1. Return the paths of transfer-ins and transfer-outs via the accounts.</td></tr>
<tr><td>params</td><td><table>
<tr><td>1</td><td>threshold</td><td>64-bit Integer</td><td>threshold of deposit amount</td></tr>
<tr><td>2</td><td>ratio_threshold</td><td>float</td><td>threshold of the ratio of repay over deposit</td></tr>
<tr><td>3</td><td>start_time</td><td>DateTime</td><td>begin of the time window</td></tr>
<tr><td>4</td><td>end_time</td><td>DateTime</td><td>end of the time window</td></tr>
</table></td></tr>
<tr><td>result</td><td><table><tr><td>1</td><td>path</td><td>?</td><td>R</td><td></td></tr></table></td></tr>
<tr><td>sort</td><td><table><tr><td>1</td><td>todo</td><td>↑</td><td></td></tr></table></td></tr>
<tr><td>limit</td><td>todo</td></tr>
<tr><td>CPs</td><td>0.0</td></tr>
<tr><td>relevance</td><td>This query is money laundering with loans involved recognition in AWL.</td></tr>
</table>

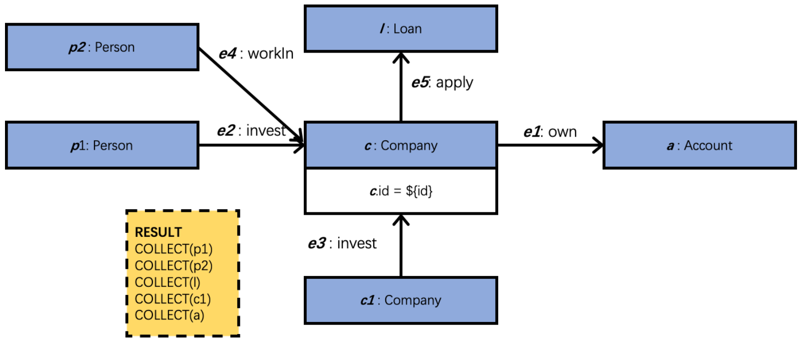
Online / read / 14

<table>
<tr><td>query</td><td colspan="4">Online / read / 14</td></tr>
<tr><td>title</td><td colspan="4">Accounts with many transfers</td></tr>
<tr><td>pattern</td><td colspan="4">
</td></tr>
<tr><td>desc.</td><td colspan="4">Find all the accounts that has more than n1 transfer-ins and more than n2 transfer-outs whose amount exceeds threshold in a specific time range between start_time and end_time. Return all these accounts.</td></tr>
<tr><td rowspan="5">params</td><td>1</td><td>threshold</td><td>64-bit Integer</td><td>threshold of transfer amount</td></tr>
<tr><td>2</td><td>start_time</td><td>DateTime</td><td>begin of the time window</td></tr>
<tr><td>3</td><td>end_time</td><td>DateTime</td><td>end of the time window</td></tr>
<tr><td>4</td><td>n1</td><td>32-bit Integer</td><td></td></tr>
<tr><td>5</td><td>n2</td><td>32-bit Integer</td><td></td></tr>
<tr><td>result</td><td>1</td><td>COLLECT(n)</td><td>[ID]</td><td>R</td></tr>
<tr><td>sort</td><td>1</td><td>todo</td><td>↑</td><td></td></tr>
<tr><td>limit</td><td colspan="4">todo</td></tr>
<tr><td>CPs</td><td colspan="4">0.0</td></tr>
<tr><td>relevance</td><td colspan="4">This query is a typical routine review for accounts which has many transfer-ins and transfer-outs.</td></tr>
</table>

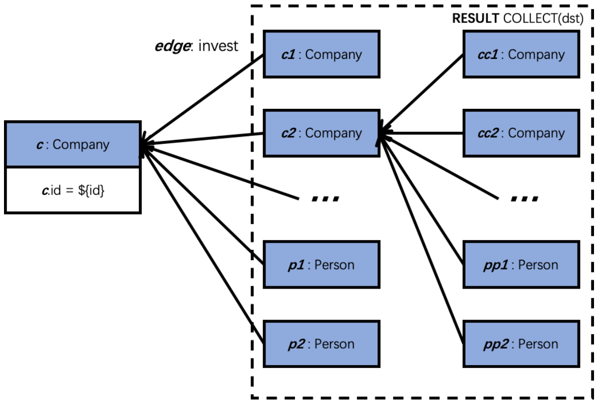
Online / read / 15

query	Online / read / 15
title	Similarity of persons who invests companies
pattern	
desc.	Given two Persons, find all the companies the two persons invest. Return the jaccard similarity between the two companies set.
params	1 | id1 | ID | id of Person1 2 | id2 | ID | id of Person2
result	1 | sim | float | R
sort	1 | todo | ↑
limit	todo
CPs	0.0
relevance	This query is a typical share holding analysis case.

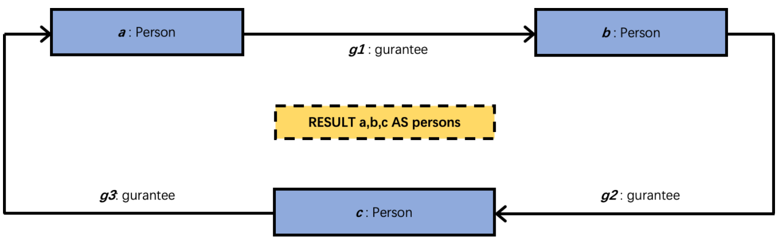
Online / read / 16

query	Online / read / 16
title	Company-related information
pattern	
desc.	Given a Company, find all the related nodes.
params	1 | `id` | ID | id of the Company
result	1 | `accounts.id` | [ID] | R 2 | `invest_persons.id` | [ID] | R 3 | `workers.id` | [ID] | R 4 | `invest_company.id` | [ID] | R 5 | `loans.id` | [ID] | R
sort	1 | todo | ↑
limit	todo
CPs	0.0
relevance	This is a company information query case.

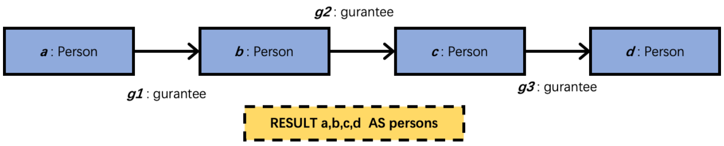
Online / read / 17

query	Online / read / 17
title	Invest relationship trace
pattern	
desc.	Given a company, find all the companies and persons the company invested by at most k steps.
params	1 | id | ID | id of the Company 2 | k | 32-bit Integer |
result	1 | COLLECT(dst) | [ID] | R
sort	1 | todo | ↑
limit	todo
CPs	0.0
relevance	This is a typical company invest relationship analysis.

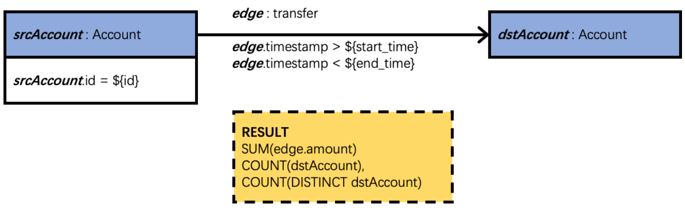
Online / read / 18

query	Online / read / 18
title	Guarantee Cycle Detection
pattern	
desc.	Find all the guarantee cycle.
result	1 | person | [ID] | R
sort	1 | todo | ↑
limit	todo
CPs	0.0
relevance	This is a typical cycle detection in guarantee.

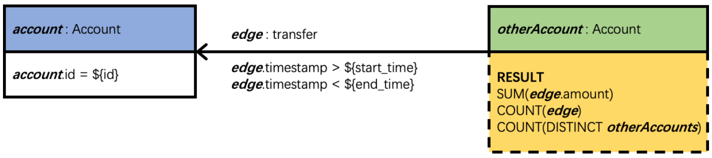
Online / read / 19

query	Online / read / 19
title	Guarantee Chain Detection
pattern	
desc.	Given an Account, find the ratio of transfer-ins from blocked Accounts in all its transfer-ins in a specific time range between start_time and end_time. Return the ratio.
result	1 | person | [ID] | R
sort	1 | todo | ↑
limit	todo
CPs	0.0
relevance	This is a typical chain detection in guarantee.

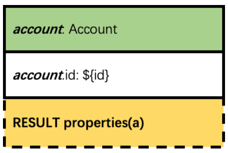
Online / read / 20

query	Online / read / 20
title	Exact Account transfer-out statistic
pattern	
desc.	Given an Account, find all the accounts the account transfers out to. Return the sum of the transfer-out amount, the count of transfer-outs, the count of distinct accounts.
params	1 | id | ID | id of the srcAccount 2 | start_time | DateTime | begin of the time window 3 | end_time | DateTime | end of the time window
result	1 | SUM(edge.amount) | 64-bit Integer | R 2 | COUNT(dstAccount) | 32-bit Integer | R 3 | COUNT(DISTINCT dstAccount) | 32-bit Integer | R
sort	1 | todo | ↑
limit	todo
CPs	0.0
relevance	This is a typical query of transfer-out statistic.

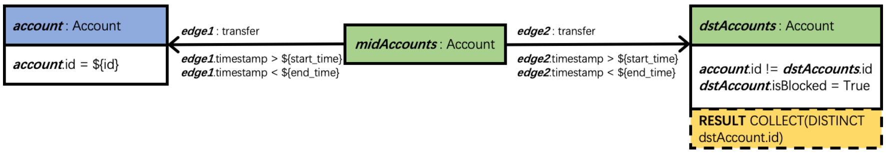
Online / read / 21

<table>
<tr><td>query</td><td>Online / read / 21</td></tr>
<tr><td>title</td><td>Exact Account transfer-in statistic</td></tr>
<tr><td>pattern</td><td>
</td></tr>
<tr><td>desc.</td><td>Given an Account, find all the accounts the account transfers in to. Return the sum of the transfer-in amount, the count of transfer-ins, the count of distinct accounts.</td></tr>
<tr><td>params</td><td><table>
<tr><td>1</td><td>id</td><td>ID</td><td>id of the Account</td></tr>
<tr><td>2</td><td>start_time</td><td>DateTime</td><td>begin of the time window</td></tr>
<tr><td>3</td><td>end_time</td><td>DateTime</td><td>end of the time window</td></tr>
</table></td></tr>
<tr><td>result</td><td><table>
<tr><td>1</td><td>SUM(edge.amount)</td><td>64-bit Integer</td><td>R</td><td></td></tr>
<tr><td>2</td><td>COUNT(edge)</td><td>32-bit Integer</td><td>R</td><td></td></tr>
<tr><td>3</td><td>COUNT(DISTINCT otherAc-count)</td><td>32-bit Integer</td><td>R</td><td></td></tr>
</table></td></tr>
<tr><td>sort</td><td><table>
<tr><td>1</td><td>todo</td><td>↑</td><td></td></tr>
</table></td></tr>
<tr><td>limit</td><td>todo</td></tr>
<tr><td>CPs</td><td>0.0</td></tr>
<tr><td>relevance</td><td>This is a typical query of transfer-in statistic.</td></tr>
</table>

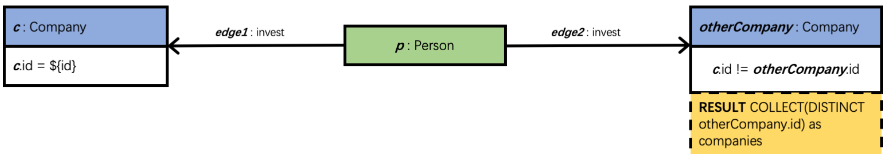
Online / read / 22

<table>
<tr><td>query</td><td>Online / read / 22</td></tr>
<tr><td>title</td><td>Exact Account properties query</td></tr>
<tr><td>pattern</td><td>
</td></tr>
<tr><td>desc.</td><td>Given an Account id, find the properties of the specific account.</td></tr>
<tr><td>params</td><td><table>
<tr><td>1</td><td>id</td><td>ID</td><td>id of the Account</td></tr>
</table></td></tr>
<tr><td>result</td><td><table>
<tr><td>1</td><td>properties</td><td></td><td>R</td><td></td></tr>
</table></td></tr>
<tr><td>sort</td><td><table>
<tr><td>1</td><td>todo</td><td>↑</td><td></td></tr>
</table></td></tr>
<tr><td>limit</td><td>todo</td></tr>
<tr><td>CPs</td><td>0.0</td></tr>
<tr><td>relevance</td><td>Property retrieve of a specific account.</td></tr>
</table>

Online / read / 23

query	Online / read / 23
title	Accounts with the same transfer sources of exact account
pattern	
desc.	Given an Account, find all the blocked accounts that connect to a third-party account which the given account has transfer-in from. Return all the accounts' id.
params	1 | `id` | ID | id of the Account 2 | `start_time` | DateTime | begin of the time window 3 | `end_time` | DateTime | end of the time window
result	1 | `COLLECT(DISTINCT dstAccount.id)` | [ID] | R
sort	1 | `todo` | ↑
limit	todo
CPs	0.0
relevance	This query is a typical analysis for gang related accounts in risk control.

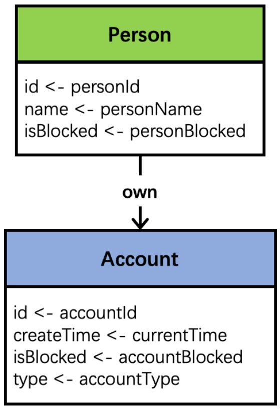
Online / read / 24

query	Online / read / 24
title	Companies with the same investor of exact company
pattern	
desc.	Given a Company, find all the companies that the investors invest at the same time.
params	1 | `id` | ID
result	1 | `companies` | [ID] | R
sort	1 | `todo` | ↑
limit	todo
CPs	0.0
relevance	This is a typical query for investor relationship analysis.

4.2 Write Queries

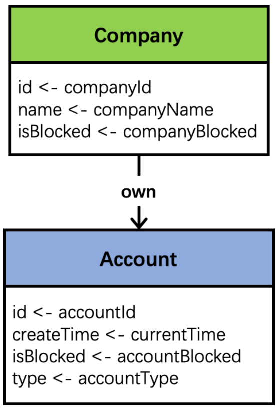
Online / write / 1

OW 1
OW 2
OW 3
OW 4
OW 5
OW 6
OW 7
OW 8
OW 9
OW 10
OW 11
OW 12
OW 13
OW 14

<table>
<tr><td>query</td><td>Online / write / 1</td></tr>
<tr><td>title</td><td>Add an Account Node owned by Person</td></tr>
<tr><td>pattern</td><td>
</td></tr>
<tr><td>desc.</td><td>Add an account node. Add a Person node and an own edge from it to the account Node.</td></tr>
<tr><td>params</td><td><table>
<tr><td>1</td><td><code>Person.personId</code></td><td>ID</td><td></td></tr>
<tr><td>2</td><td><code>Person.personName</code></td><td>String</td><td></td></tr>
<tr><td>3</td><td><code>Person.personBlocked</code></td><td>Boolean</td><td></td></tr>
<tr><td>4</td><td><code>Account.accountId</code></td><td>ID</td><td></td></tr>
<tr><td>5</td><td><code>Account.currentTime</code></td><td>DateTime</td><td></td></tr>
<tr><td>6</td><td><code>Account.accountBlocked</code></td><td>Boolean</td><td></td></tr>
<tr><td>7</td><td><code>Account.accountType</code></td><td>String</td><td></td></tr>
</table></td></tr>
<tr><td>CPs</td><td>0.0</td></tr>
</table>

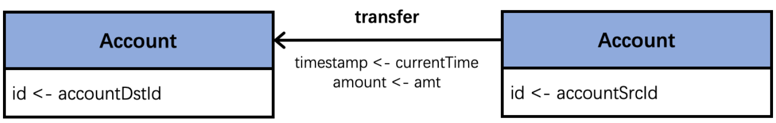
Online / write / 2

OW 1 OW 2 OW 3 OW 4 OW 5 OW 6 OW 7 OW 8 OW 9 OW 10 OW 11 OW 12 OW 13 OW 14

<table>
<tr><td>query</td><td>Online / write / 2</td></tr>
<tr><td>title</td><td>Add an Account Node owned by Company</td></tr>
<tr><td>pattern</td><td>
</td></tr>
<tr><td>desc.</td><td>Add an account node. Add a Company node and an own edge from it to the account Node.</td></tr>
<tr><td>params</td><td><table>
<tr><td>1</td><td>Company.companyId</td><td>ID</td><td></td></tr>
<tr><td>2</td><td>Company.companyName</td><td>String</td><td></td></tr>
<tr><td>3</td><td>Company.companyBlocked</td><td>Boolean</td><td></td></tr>
<tr><td>4</td><td>Account.accountId</td><td>ID</td><td></td></tr>
<tr><td>5</td><td>Account.currentTime</td><td>DateTime</td><td></td></tr>
<tr><td>6</td><td>Account.accountBlocked</td><td>Boolean</td><td></td></tr>
<tr><td>7</td><td>Account.accountType</td><td>String</td><td></td></tr>
</table></td></tr>
<tr><td>CPs</td><td>0.0</td></tr>
</table>

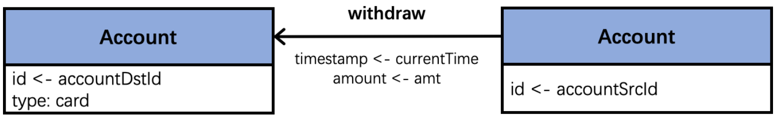
Online / write / 3

OW 1 OW 2 OW 3 OW 4 OW 5 OW 6 OW 7 OW 8 OW 9 OW 10 OW 11 OW 12 OW 13 OW 14

<table>
<tr><td>query</td><td>Online / write / 3</td></tr>
<tr><td>title</td><td>Add transfer between accounts</td></tr>
<tr><td>pattern</td><td>
</td></tr>
<tr><td>desc.</td><td>Add a transfer edge from an account node to another</td></tr>
<tr><td>params</td><td><table>
<tr><td>1</td><td>transfer.timestamp</td><td>DateTime</td><td></td></tr>
<tr><td>2</td><td>transfer.amount64-bit</td><td>Integer</td><td></td></tr>
</table></td></tr>
<tr><td>CPs</td><td>0.0</td></tr>
</table>

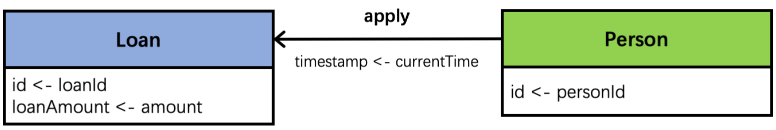
Online / write / 4

OW 1 OW 2 OW 3 OW 4 OW 5 OW 6 OW 7 OW 8 OW 9 OW 10 OW 11 OW 12 OW 13 OW 14

<table>
<tr><td>query</td><td>Online / write / 4</td></tr>
<tr><td>title</td><td>Add withdraw between accounts</td></tr>
<tr><td>pattern</td><td>
</td></tr>
<tr><td>desc.</td><td>Add a withdraw edge from an account node to another account node whose type is card</td></tr>
<tr><td>params</td><td><table>
<tr><td>1</td><td>Account.accountSrcId</td><td>ID</td><td></td></tr>
<tr><td>2</td><td>Accout.accountDstId</td><td>ID</td><td></td></tr>
<tr><td>3</td><td>withdraw.timestamp</td><td>DateTime</td><td></td></tr>
<tr><td>4</td><td>withdraw.amount</td><td>64-bit Integer</td><td></td></tr>
</table></td></tr>
<tr><td>CPs</td><td>0.0</td></tr>
</table>

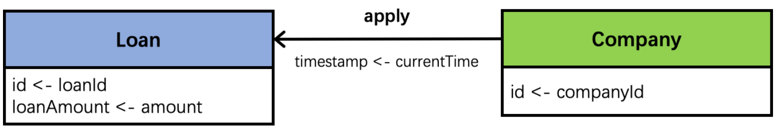
Online / write / 5

OW 1 OW 2 OW 3 OW 4 OW 5 OW 6 OW 7 OW 8 OW 9 OW 10 OW 11 OW 12 OW 13 OW 14

<table>
<tr><td>query</td><td>Online / write / 5</td></tr>
<tr><td>title</td><td>Add Loan applied by Person</td></tr>
<tr><td>pattern</td><td>
</td></tr>
<tr><td>desc.</td><td>Add a Loan Node and add an apply edge from a person node to it.</td></tr>
<tr><td>params</td><td><table>
<tr><td>1</td><td>Loan.loanId</td><td>ID</td><td></td></tr>
<tr><td>2</td><td>Loan.loanAmount</td><td>64-bit Integer</td><td></td></tr>
<tr><td>3</td><td></td><td>ID</td><td></td></tr>
</table></td></tr>
<tr><td>CPs</td><td>0.0</td></tr>
</table>

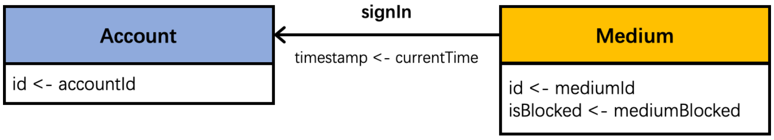
Online / write / 6

OW 1 OW 2 OW 3 OW 4 OW 5 OW 6 OW 7 OW 8 OW 9 OW 10 OW 11 OW 12 OW 13 OW 14

<table>
<tr><td>query</td><td>Online / write / 6</td></tr>
<tr><td>title</td><td>Add Loan applied by Company</td></tr>
<tr><td>pattern</td><td>
</td></tr>
<tr><td>desc.</td><td>Add a Loan Node and add an apply edge from a company node to it.</td></tr>
<tr><td>params</td><td><table>
<tr><td>1</td><td>Loan.loanId</td><td>ID</td><td></td></tr>
<tr><td>2</td><td>Loan.loanAmount</td><td>64-bit Integer</td><td></td></tr>
<tr><td>3</td><td>Company.companyId</td><td>ID</td><td></td></tr>
</table></td></tr>
<tr><td>CPs</td><td>0.0</td></tr>
</table>

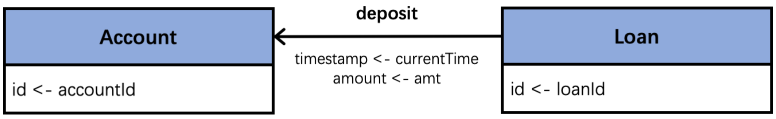
Online / write / 7

OW 1 OW 2 OW 3 OW 4 OW 5 OW 6 OW 7 OW 8 OW 9 OW 10 OW 11 OW 12 OW 13 OW 14

query	Online / write / 7
title	Account signed in with Medium
pattern	
desc.	Add an Medium node and add a signIn edge from it to an Account node
params	1 `Account.accountId` ID 2 `Medium.mediumId` ID 3 `Medium.mediumBlocked` Boolean 4 `signIn.currentTime` DateTime
CPs	0.0

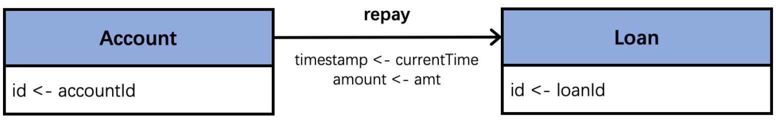
Online / write / 8

OW 1 OW 2 OW 3 OW 4 OW 5 OW 6 OW 7 OW 8 OW 9 OW 10 OW 11 OW 12 OW 13 OW 14

query	Online / write / 8
title	Loan Deposit
pattern	
desc.	Add a deposit edge from a Loan node to an Account node
params	1 `Account.accountId` ID 2 `Loan.loanId` ID 3 `deposit.currentTime` DateTime 4 `deposit.amt` 64-bit Integer
CPs	0.0

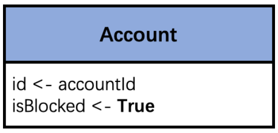
Online / write / 9

OW 1 OW 2 OW 3 OW 4 OW 5 OW 6 OW 7 OW 8 OW 9 OW 10 OW 11 OW 12 OW 13 OW 14

query	Online / write / 9
title	Loan repay
pattern	
desc.	Add a repay edge from an Account node to a Loan node
params	1 `Account.accountId` ID 2 `Loan.loanId` ID 3 `repay.currentTime` DateTime 4 `repay.amt` 64-bit Integer
CPs	0.0

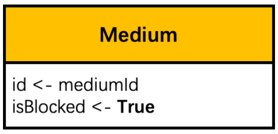
Online / write / 10

OW 1 OW 2 OW 3 OW 4 OW 5 OW 6 OW 7 OW 8 OW 9 OW 10 OW 11 OW 12 OW 13 OW 14

query	Online / write / 10
title	Block an account of high risk
pattern	
desc.	Set an account's isBlocked to True.
params	1 `Account.accountId` ID
CPs	0.0

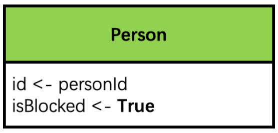
Online / write / 11

OW 1 OW 2 OW 3 OW 4 OW 5 OW 6 OW 7 OW 8 OW 9 OW 10 OW 11 OW 12 OW 13 OW 14

query	Online / write / 11
title	Block a medium of high risk
pattern	
desc.	Set a medium's isBlocked to True
params	1 `Medium.accountId` ID
CPs	0.0

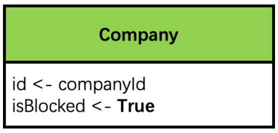
Online / write / 12

OW 1 OW 2 OW 3 OW 4 OW 5 OW 6 OW 7 OW 8 OW 9 OW 10 OW 11 OW 12 OW 13 OW 14

query	Online / write / 12
title	Block a person of high risk
pattern	
desc.	Set a person's isBlocked to True.
params	1 `Person.personId` ID
CPs	0.0

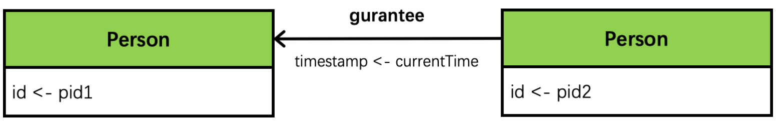
Online / write / 13

OW 1 OW 2 OW 3 OW 4 OW 5 OW 6 OW 7 OW 8 OW 9 OW 10 OW 11 OW 12 OW 13 OW 14

query	Online / write / 13
title	Block a company of high risk
pattern	
desc.	Set a company's isBlocked to True.
params	1 · `Company.companyId` · ID
CPs	0.0

Online / write / 14

OW 1 OW 2 OW 3 OW 4 OW 5 OW 6 OW 7 OW 8 OW 9 OW 10 OW 11 OW 12 OW 13 OW 14

query	Online / write / 14
title	Add guarantee between persons
pattern	
desc.	Add a guarantee edge from a person node to another person node
params	1 · `guarantee.timestamp` · DateTime 2 · `Person.id1` · ID 3 · `Person.id2` · ID
CPs	0.0

4.3 Read-Write Queries

Online / read-write / 1

ORW 1 ORW 2 ORW 3 ORW 4 ORW 5

query	Online / read-write / 1
title	High risk account blocked after frequent Money-Laundering transfer cycle detected
compose.	This read-write query contains the reads and writes below, • Online / Read / 22 • Online / Write / 3 • Online / Read / 4 • Online / Write / 10
desc.	With the reads and writes, this query works as: * With Online / Read / 22, blocked status of related account is read. * With Online / write / 3, many transfer edges are inserted. And with enough transfer edges inserted, a transfer cycle via a third-party is formed. * With Online / read / 4, a three accounts cycle is detected and the sum of transferred fund amount is calculated. * When the amount sum exceeds a threshold, Online / write / 10 is triggered to mark all the account in the cycle as blocked.
params	1	threshold of amount sum	Float
relevance	It is a typical AWL case in risk control.

Online / read-write / 2

ORW 1 ORW 2 ORW 3 ORW 4 ORW 5

query	Online / read-write / 2
title	High risk account blocked after many transfer-outs from blocked account
compose.	This read-write query contains the reads and writes below, • Online / Read / 22 • Online / Write / 3 • Online / Read / 6 • Online / Write / 10
desc.	With the reads and writes, this query works as: * With Online / Read / 22, blocked status of related account is read. * With Online / write / 3, many transfer edges are inserted. Then enough transfer-out edges from the blocked and unblocked accounts are inserted to the target account. * With Online / read / 6, many transfer-outs is detected and the ratio of the account transfer-ins from blocked account in all transfer-ins is calculated. * When the ratio exceeds a threshold, Online / write / 10 is triggered to mark the account as blocked.
params	1	threshold of ratio	Float
relevance	It is a typical fraud detection case in risk control.

Online / read-write / 3

ORW 1 ORW 2 ORW 3 ORW 4 ORW 5

query	Online / read-write / 3
title	High risk account mid-account blocked after fast-in and fast-out detected
compose.	This read-write query contains the reads and writes below, • Online / Read / 22 • Online / Write / 3 • Online / Read / 10 • Online / Write / 10
desc.	With the reads and writes, this query works as: * With Online / Read / 22, blocked status of related account is read. * With Online / write / 3, many transfer edges are inserted. Then enough transfer-out edges from the blocked and unblocked accounts are inserted to the target account. * With Online / read / 10, the pattern of Fast-in and Fast-out is detected and the count of the account transfer-ins and transfer-outs is calculated. * When the count exceeds a threshold, Online / write / 10 is triggered to mark the account as blocked.
params	1 | threshold of transfer-ins and transfer-outs count. | Float
relevance	It is a typical fraud detection and AWL case in risk control.

Online / read-write / 4

ORW 1 ORW 2 ORW 3 ORW 4 ORW 5

query	Online / read-write / 4
title	High risk account blocked after frequent big transfers
compose.	This read-write query contains the reads and writes below, • Online / Read / 22 • Online / Write / 3 • Online / Read / 12 • Online / Write / 10
desc.	With the reads and writes, this query works as: * With Online / Read / 22, blocked status of related account is read. * With Online / write / 3, many transfer edges are inserted. Then enough transfer-out edges from the blocked and unblocked accounts are inserted to the target account. * With Online / read / 12, the pattern of abnormal transfer from the target account is detected and the count of the big transfers from the account is calculated. * When the count exceeds a threshold, Online / write / 10 is triggered to mark the account as blocked.
params	1 | threshold of the big transfers. | Float
relevance	It is a typical case in risk control.

ORW 1
ORW 2
ORW 3
ORW 4
ORW 5

Online / read-write / 5

query	Online / read-write / 5
title	High risk person blocked after guarantee cycle detected
compose.	This read-write query contains the reads and writes below, • Online / Write / 14 • Online / Read / 18 • Online / Write / 12
desc.	With the reads and writes, this query works as: * With Online / write / 14, many guarantee edges are inserted. Then enough guarantee edges are inserted to form a guarantee cycle. * With Online / read / 18, the pattern of a guarantee cycle is detected and all the persons in the cycle is returned. * Because of the high risk of guarantee cycle, Online / write / 12 is triggered to mark all the person as blocked.
relevance	It is a typical fraud detection in risk control.

5 Nearline Workload

[Future Work]

6 Offline Workload

[Future Work]

7 Auditing Rules

TODO

8 Related Work

TODO

A Choke Points

Introduction

Choke points are a superset of [**LdbcDeliverable**] with the exception of CP 7.1, which was removed and replaced with a new choke point. The correlations between choke points and queries are displayed in **??**.

A.1 Path Filtering

CP-1.1: [TBFE] Timestamp-based and amount-based filter on edges `TPC–H 1.1`

Almost all the edges has property Timestamp and the transfer edges has property Amount. Timestamp and amount property is used in filtering.

CP-1.2: [RFBD] Recursive Filtering with backward dependency in variable-length paths `TPC–H 1.2`

In some cases, there are some special pattern that is typical in financial scenarios. Take Online/read/11 for example. In this query, it is supposed there are paths from the start Account Node to other Account Node at most 3 steps via Transfer relationship. These paths should meet these requirements or patterns:

- All money-transfer paths A ->[e1]->B-[e2]->...->X in which the timestamp of each transfer ei is larger than that of ei-1
- All money-transfer paths A ->[e1]->B-[e2]->...->X in which the amount of each transfer ei is smaller than that of ei-1

These cases contain this pattern: Online / read / [1,2,3,4,7,8,9,10,11,13]

CP-1.3: [RPQ] Regular Path Query `TPC–H 1.3`

In some cases, the edges may be of different type in variant-length paths, e.g., paths like A ->[e1]->B-[e2]->...->X in which the type of ei is transfer and the type of ei-1 is withdraw. When the types varies, it is supposed to support query using regular expressions. Examples cases: Online / read / 11

A.2 Complex Pattern

CP-2.1: [] Cycle Detection `TPC–H 2.1`

CP-2.2: [] Chain Detection `TPC–H 2.2`

A.3 Read-write query

CP-3.1: [] Real-time queries composed of reads and writes `TPC–H 3.1`

A.4 Complex Writes

CP-4.1: [] Multiplicate edges insert `TPC–H 4.1`

CP-4.2: [] Property level update `TPC–H 4.2`

A.5 Aggregation Performance

CP-5.1: [] Complex aggregations `TPC–H 5.1`