Use of FpML for Risk Evaluation

Marc Gratacos (ISDA) & Karel Engelen (ISDA)

December 2011

Abstract

The objective of this paper is to show how querying trades in FpML format allows to perform risk evaluation of Over-The-Counter (OTC) derivatives. We evaluated recent regulatory consultation papers from the international regulatory community in search for specific regulatory questions around risk evaluation that we can translate into data queries. We implement the queries using a sample FpML data set and a widely used XML query language (XQuery) to extract the relevant information. We adapt certain queries to make them relevant for the test data set.

While the queries would require certain enhancements for implementation outside of the test environment, the data - in FpML format – is sufficiently structured to retrieve the required information in an unambiguous way. When the questions are clearly defined, implementing the queries has shown to be fairly straightforward.

Introduction

The changes in the US regulatory framework for OTC derivatives are providing regulators access to enormous amounts of data in electronic format. Every OTC derivative transaction, whether centrally cleared or not needs to be reported to a Swap Data Repository (SDR). FpML is becoming the *defacto* standard for OTC trade information delivery to the SDRs, thus making the data available in a structured way. The format in which data gets reported to the SDRs is part of the FpML standard and publicly available at no cost. In addition, the OTC derivatives industry is developing Taxonomies and Unique Product Identifiers (UPI) that will assist the regulators with the classification of OTC derivative trades for the different asset classes[1].

Questions have been raised, how the set of data available to the regulators can be analyzed and in particular, if the current information, tools and technology allow answering the regulatory questions or, if an additional layer needs to be defined to structure the data and make it meaningful and suitable for querying.

To address these questions we built a database using publicly available examples of OTC contracts. We extracted a set of specific questions raised in the Canadian Securities Administrators CSA Consultation Paper 91-402 on Derivatives Trade Repositories[2] and defined the queries to find the answers to those questions. We implemented the queries using existing and widely used technology and tools.

FpML – Financial products Markup Language

FpML[3] is the XML standard for Over-The-Counter (OTC) derivatives, following W3C rules. The standard is freely available with an open source license, developed by the OTC derivatives industry, and managed by the International Swaps and Derivatives Association, Inc. (ISDA). In addition to XML Schema files, validation rules, and documentation, ISDA publishes a set of examples showing a spectrum of products and messages represented in FpML for each version.

Framework

We undertake an exercise to analyze a sample set of regulatory questions and run them against a database with test data. More specifically, we answer the questions raised in section 3.4 (c) of the Canadian Securities Administrators CSA Consultation Paper 91-402 on Derivatives Trade Repositories.

As dataset we use the FpML examples that are part of the public standard and are typically based on actual trade confirmations. For this exercise, we use the set of FpML examples available in version 5.3[4].

[1] http://www2.isda.org/attachment/MzQyMA==/ISDA%20UPI%20Implementation%20Plan%20Final.pdf
[2] http://www.gov.ns.ca/nssc/docs/derivativespaperJune_23_2011.pdf
[3] http://www.fpml.org/
[4] http://www.fpml.org/spec/fpml-5-3-1-wd-1/

Technically, we perform the following basic software set up to create an environment to effectively perform the queries on the data:

eXist XML Database

An XML database called eXist-db[5] is installed locally to store all the FpML examples. eXist-db is an open source database management system built using XML technology. It stores XML data according to the XML data model and features efficient, index-based XQuery processing.

Once eXist is up and running, the default eXist Client Shell is used to run the implemented queries.

XQuery Editor

oXygen XML Editor[6] is the editor used to develop the XQuery queries. It is a multi-platform XML editor, XSLT/XQuery debugger and profiler with Unicode support. It is a Java application, so it can run in Windows, Mac OS X, and Linux.

Performance

The scope of this work does not include the performance aspect when applying this type of querying to large sets of data. While XML technologies and databases are used in this context in the financial industry[7][8] and other industries, this would need to be considered as part of a separate thread.

Regulatory Questions

As mentioned above, we use the questions raised in the Canadian Securities Administrators CSA Consultation Paper 91-402 on Derivatives Trade Repositories. These questions are listed below. However, to make the questions more relevant for the test data set, we apply certain changes. As an example: in the test data set most example trades are in Euro (EUR) or US dollar (USD), so we replace Canadian dollar (CAD) in the original question with USD. Similarly, for certain of the queries we define a product subset if the question is more general.

For each question we indicate the changes we make and the rationale for the changes. We believe that none of these changes are critical for the type of analysis regulators want to perform.

List of nine questions raised in the CSA Consultation Paper:

1. Aggregate notional data for all contracts traded or settled in Canadian dollars, including a breakdown by reference entity and/or sector.
2. A list of the top counterparties trading Canadian dollar denominated contracts with each counterparty's aggregate notional position and aggregate position by contract type.

[5] http://exist.sourceforge.net/
[6] http://www.oxygenxml.com/
[7] http://www.ibm.com/developerworks/wikis/download/attachments/1824/Derivatives+Trading+DB2+XML.pdf
[8] http://blogs.wsj.com/venturecapital/2011/10/27/big-data-success-stories-marklogic/

3. A list of the top counterparty positions for each of the largest financial groups in Canada.
4. Aggregate notional data for contracts written on Canadian-domiciled corporations (reference entities), including a list of the top aggregate notional counterparty positions for contracts written on each firm.
5. A list of the top counterparties' aggregate notional positions where the contract references the debt of the government of Canada.
6. A list of top counterparties' aggregate notional positions where the contract references a specific commodity.
7. A list of the top counterparties' aggregate notional positions where contracts reference the debt of one of the ten largest Canadian financial groups.
8. Data on the overall level of activity of each of the Canadian banks in each asset class.
9. Each of the Canadian bank's overall positions in specific products within an asset class.

Implementation

We implemented the questions in XQuery. The full set of queries is available in **Annex A**.

Question 1: Aggregate notional data for all contracts traded or settled in Canadian dollars, including a breakdown by reference entity and/or sector

Changes to Question 1

- The currency is changed to USD in all query implementations since most of the FpML examples are in USD and EUR. Rationale: make the question more relevant for the test data available.
- We limited the product scope of this query implementation to Single Name Credit Default Swap, Credit Index, Return Swap, and bespoke products. Rationale: This limited product coverage is done to show multiple products but keep the implementation simple. As a consequence the breakdown is implemented by reference entity, credit index, or equity asset. However, the implementation can be easily expanded to support additional product types and underlying assets. The work on taxonomies and UPIs can be a guide in defining the product categories for inclusion in the query.

Generic Product

We include the support for bespoke products in this question through the FpML Generic Product structure. The Generic Product provides a placeholder representation with a few key identifying details to allow reporting of products that aren't otherwise able to be represented in FpML. The use of the Generic Product within the query implementation is equivalent to the other products.

Implementation of Question 1

This question is executed in four separate queries, one for each product in scope. The separate queries allow for an easier implementation. Expanding the product scope would lead to additional queries.

A default namespace is defined. In this case, the samples we use belong to the FpML confirmation view but this could be applied to other views as well.

```
declare default element namespace "http://www.fpml.org/FpML-5/reporting";
```

First, the query looks for all CDS contracts traded or settled in USD

```
let $trade := collection()//trade[creditDefaultSwap/protectionTerms/calculationAmount/currency
= 'USD' or creditDefaultSwap/cashSettlementTerms/settlementCurrency= 'USD']
```

Then the query looks for all unique reference entities and it creates a loop to group the ones with the same id:

```
for $entityId in distinct-
values($trade/creditDefaultSwap/generalTerms/referenceInformation/referenceEntity/entityId)
let $byEntityId :=
$trade/creditDefaultSwap/generalTerms/referenceInformation/referenceEntity[entityId=$entityId]
```

For all trades with the same reference entity, the following code aggregates their notional amount.

```
return
<creditEntityExposure>
        <entityId>{$entityId}</entityId>
        <aggregateNotional>{sum($byEntityId/../../../protectionTerms/calculationAmount/amount
)}</aggregateNotional>
</creditEntityExposure>,
```

For bespoke trades, the query looks for the FpML genericProduct product element:

```
let $trade := collection()//trade[genericProduct/notional/currency = 'USD' or
genericProduct/settlementCurrency = 'USD']
```

The structure of the query is the same as the other products.

Output

As the snippet below shows, the output provides a simple XML fragment with each entityId and aggregate trade notional for each reference entity.

Snippet of the generated output:

```
<creditEntityExposure>
   <entityId>GG3682</entityId>
   <aggregateNotional>1.0E7</aggregateNotional>
</creditEntityExposure>
<creditEntityExposure>
   <entityId>0C575S</entityId>
   <aggregateNotional>1.0E7</aggregateNotional>
</creditEntityExposure>
…
```

The same query structure and output is developed for Credit Indices, Return Swaps (with an underlying equity), and bespoke trades. The following snippet shows the output for the Credit

Index exposure in USD:

```
<creditIndexExposure>
   <indexName>Dow Jones CDX NA IG.2</indexName>
   <aggregateNotional>2.5E7</aggregateNotional>
</creditIndexExposure>
<creditIndexExposure>
   <indexName>Dow Jones iTraxx Europe Consumers Series 2 Version 1</indexName>
   <aggregateNotional>5.0E7</aggregateNotional>
</creditIndexExposure>
…
```

And the following for bespoke trades' exposure in USD:

```
…
<bespokeExposure>
   <instrumentId>X234123</instrumentId>
   <aggregateNotional>1.0E7</aggregateNotional>
</bespokeExposure>
…
```

Note: Since the output is an XML snippet, this can be further subjected to additional XQuery interrogation to answer specific facets of a question, particularly if the resulting output is large.

Question 2: A list of the top counterparties trading Canadian dollar denominated contracts with each counterparty's aggregate notional position and aggregate position by contract type

Changes to Question 2 and their rationale

As described in Question 1, the currency is changed to USD. The product scope of this query implementation is limited to Single Name Credit Default Swap, Return Swap, and Interest Rate Swap but other products could be added easily.

Implementation of Question 2

This question is divided in two parts, the notional aggregate by product and the total notional amount. The query creates collections for USD trades for each product type. This is critical in order to calculate the aggregate notional by product type, which is done using the XQuery sum function.

The total notional aggregates the notional amounts of all USD trades by party. As defined in the question, the parties' aggregates are ordered by total exposure descending

Output

The output shows the aggregate exposure (sum of trade notional amounts) for single name credit default swaps, return swaps, and interest rate swaps by party (identified with party Id). In addition, the total exposure of all these three products by party is also provided.

Snippet of the generated output:

```
…
<totalExposureByParty>
   <partyId>ABCBICXXX</partyId>
   <creditExposure>2.068E8</creditExposure>
   <returnSwapExposure>2.8469376E7</returnSwapExposure>
   <interestRateSwapExposure>0</interestRateSwapExposure>
```

```
   <totalExposure>2.35269376E8</totalExposure>
</totalExposureByParty>
…
```


Question 3: A list of the top counterparty positions for each of the largest financial groups in Canada

Changes to Question 3 and their rationale

Currently there is no mechanism in FpML to know whether the party to the trade is from a specific country. This issue could be solved with a look up to a party database containing the country information or with the introduction of the Legal Entity Identifier (LEI), which will incorporate the country of the institution[9]. As a consequence, the implementation of the query lists all party positions available in the database, not only the Canadian firms, grouped by product and showing the higher positions on top of the list.

The product scope of the implementation is limited to Credit Default Swap and Index, Return Swap, and Interest Rate Swap. However, the structure of the query is the same for all products and the implementation could be easily extended to support additional products.

Implementation of Question 3

The question is divided in three independent queries, one covering both Single Name and Index CDS and the other two for Return Swap and Interest Rate Swap. The queries group all trades by parties and then get the trade Ids and notional for each party. Finally, it orders the positions by their notional amount.

Output

The output shows all credit, return swap, and interest rate swap positions, represented by trade id and notional, for each party.

Snippet of the generated output:


```
<creditPositionsByParty>
  <partyId>XYZBICXXX</partyId>
    <position>
        <tradeId tradeIdScheme="http://www.swapswire.com/spec/2001/trade-id-1-
0">37209</tradeId>
        <notional>
            <currency>JPY</currency>
            <amount>500000000.0</amount>
        </notional>
    </position>
    <position>
        <tradeId tradeIdScheme="http://www.xyzbank.com/cd-trade-id">xyz1234</tradeId>
        <notional>
            <currency>JPY</currency>
            <amount>500000000</amount>
        </notional>
    </position>
…

<returnSwapPositionsByParty>
    <partyId>HEGDUS33</partyId>
```


[9] FpML will support the use of LEI, at the time of writing of this paper LEI codes are not yet available..

```
  <position>
    <tradeId tradeIdScheme="http://www.abc.com/swaps/trade-id">TRS-01</tradeId>
    <notional>
      <currency>USD</currency>
      <amount>28469376</amount>
    </notional>
  </position>
  <position>
    <tradeId tradeIdScheme="http://www.abc.com/swaps/trade-id">TRS-02</tradeId>
    <notional>
      <currency>EUR</currency>
      <amount>19785157.16</amount>
    </notional>
  </position>
</returnSwapPositionsByParty>
```


…

Question 4: Aggregate notional data for contracts written on Canadian domiciled corporations (reference entities), including a list of the top aggregate notional counterparty positions for contracts written on each firm

Changes to Question 4 and their rationale

As mentioned for question 3, currently there is no mechanism in FpML to know whether the reference entity is from a specific country. This issue could be solved with a look up to a reference entity database containing such information or it may be solved when the Legal Entity Identifier (LEI) is introduced. In this particular case, the product scope of the implementation is limited to Credit Default Swap.

Implementation of Question 4

The query does not distinguish between Canadian and non-Canadian reference entities. A separate database lookup to know the country of the reference entity would be necessary.

The query is retrieving the trade Id of the first partyTradeIdentifier block. This simplifies the query for the purpose of this paper but in a real implementation the reference mechanism should be supported in order to get the exact partyTradeIdentifier block for each party.

Output

The output shows all credit positions, represented by trade id and notional, for each reference entity (identified by entity id) and grouped by currency. In addition, the sum of all positions' notional for each reference entity is reported by currency in the aggregate notional structure.


```
<creditPositionsByEntityId>
        <entityId>004CC9</entityId>
        <creditPositionsByCurrency>
                <aggregateNotional>
                        <currency>JPY</currency>
                        <amount>5.0E8</amount>
                </aggregateNotional>
                <position>
                        <tradeId tradeIdScheme="http://www.swapswire.com/spec/2001/trade-
```

id-1-0">37209</tradeId>
<amount>500000000.0</amount>
</position>
</creditPositionsByCurrency>
<creditPositionsByCurrency>
<aggregateNotional>
<currency>**USD**</currency>
<amount>1.0E7</amount>
</aggregateNotional>
<position>
<tradeId tradeIdScheme="http://www.swapswire.com/spec/2001/trade-id-1-0">37258</tradeId>
<amount>10000000.0</amount>
</position>
</creditPositionsByCurrency>
</creditPositionsByEntityId>
...

Question 5: A list of the top counterparties' aggregate notional positions where the contract references the debt of the government of Canada

Changes to Question 5 and their rationale

Similarly to what happened in Q3 and Q4, currently, there is no mechanism in FpML to know whether the debt is from a specific country by looking at the instrument identifier. However, for this question we make a few assumptions:

- For debt from the government of Canada, the issuer of the bond is 'Her Majesty in right of Canada'.
- For credit default swaps (CDS) trades referencing the debt of the government of Canada, we assume that the reference entity of the CDS is the government of Canada. The reference entity of the government of Canada may be identified by the RED pair clip '27CBJG' and/or the entity name 'Canada'.
- The product scope of the implementation for this question is limited to Credit Default Swap.

Implementation of Question 5

The query distinguishes between Canadian and non-Canadian debt by analyzing:

- The issuer of the bond and looking whether it contains the value 'Her Majesty in right of Canada'.
- The entity id and whether it matches the value '27CBJG'.
- The entity name and whether it matches the value 'Canada'.

Output

The output shows the credit exposure to the debt of the government of Canada by party. In order to properly compute the aggregate exposures, they are grouped by currency.

<creditExposureByParty>
<partyId>XYZBICXXX</partyId>

```
        <creditExposureByCurrency>
                <aggregateNotional>
                        <currency>USD</currency>
                        <amount>1.0E7</amount>
                </aggregateNotional>
        </creditExposureByCurrency>
</creditExposureByParty>
```


Question 6: A list of the top counterparties' aggregate notional positions where the contract references a specific commodity

Changes to Question 6 and their rationale

The product scope covers financial commodities. In order to get the notional amounts for physical commodities, for example electricity, the query would be more complex to aggregate the delivery amounts that may change in each period. For financial commodities there exists a Total Notional Quantity element that facilitates retrieving the total amount for each transaction.

Implementation of Question 6

The question is divided in four queries, one for each commodity asset. In order to find the type of asset, each query looks for the Commodity Reference Price code in the instrument Id element and whether it contains the substring ELECTRICITY, GAS, COAL, or OIL. The Commodity Reference Price code is an ISDA defined list and FpML published it as a scheme list[10].

Output

The output shows the aggregate trade notionals for power, gas, coal, and oil financial commodity swaps for each party.


```
<commodityExposureByParty>
   <partyId>Party_A_BIC_CODE</partyId>
   <powerExposure>1.097552E6</powerExposure>
</commodityExposureByParty>
<commodityExposureByParty>
   <partyId>Party_B_BIC_CODE</partyId>
   <powerExposure>1.097552E6</powerExposure>
</commodityExposureByParty>
<commodityExposureByParty>
   <partyId>Party_A_BIC_CODE</partyId>
   <gasExposure>3.1025E6</gasExposure>
</commodityExposureByParty>
…
```


Question 7: A list of the top counterparties' aggregate notional positions where the contracts reference the debt of one of the ten largest Canadian financial groups

[10] http://www.fpml.org/coding-scheme/commodity-reference-price-1-0.xml

Changes to Question 7 and their rationale

Currently, there is no mechanism in FpML to know whether the debt is from a top financial group in a specific country. This issue could be solved with a look up to a bond database containing such information. The product scope of the implementation is limited to Credit Default Swap.

Implementation of Question 7

The implementation is similar to Question 5 but in this case the query is looking for debt from multiple institutions so a database or file lookup would be a must to identify the institutions which debt needs to be taken into account.

There is a separate aggregate notional structure for each currency so trades are aggregated for the same currency.

Output

The output shows the aggregate trade notional amounts for single name credit default swaps with a Canadian bond for each party.

```
<creditExposureByParty>
        <partyId>XYZBICXXX</partyId>
        <creditExposureByCurrency>
                <aggregateNotional>
                        <currency>JPY</currency>
                        <amount>1.0E9</amount>
                </aggregateNotional>
        </creditExposureByCurrency>
        <creditExposureByCurrency>
                <aggregateNotional>
                        <currency>EUR</currency>
                        <amount>4.0E8</amount>
                </aggregateNotional>
        </creditExposureByCurrency>
</creditExposureByParty>
```

Question 8: Data on the overall level of activity of each of the Canadian banks in each asset class

Changes to Question 8 and their rationale

The query does not filter by Canadian banks only so all banks are included. With the introduction of the Legal Entity Identifier (LEI), this could be achieved by looking at the partyId with an LEI value.

The product scope of this query implementation is limited to Single Name Credit Default Swap, Return Swap, and Interest Rate Swap but other products could be added easily.

Implementation of Question 8

This question is divided in two parts, firstly it groups the trades by party id and then it aggregates by asset class and currency.

Output

The output shows the total exposure (sum of trade notional amounts) for credit, equity, and rates assets by party and currency.

…
<totalExposureByParty>
  <partyId>XYZBICXXX</partyId>
  <creditExposureByCurrency>
    <currency>JPY</currency>
    <aggregateNotional>1.5E9</aggregateNotional>
  </creditExposureByCurrency>
  <creditExposureByCurrency>
    <currency>USD</currency>
    <aggregateNotional>2.068E8</aggregateNotional>
  </creditExposureByCurrency>
  <creditExposureByCurrency>
    <currency>EUR</currency>
    <aggregateNotional>2.0E7</aggregateNotional>
  </creditExposureByCurrency>
  <ratesExposureByCurrency>
    <currency>MXN</currency>
    <aggregateNotional>1.0E8</aggregateNotional>
  </ratesExposureByCurrency>
</totalExposureByParty>
…

Question 9: Each of the Canadian bank's overall positions in specific products within an asset class

Changes to Question 9 and their rationale

As happens with Question 8 the query does not filter by Canadian banks only. With the introduction of the Legal Entity Identifier (LEI), this could be achieved by looking at the partyId with an LEI value.

The product scope of this query implementation is limited to Credit Derivatives: Single Name Credit Default Swap, Credit Index, Credit Basket, and Credit Option. Again, additional asset classes and products could be added easily.

Implementation of Question 9

This question is divided in multiple parts, one for each product type. For each product and party, a total notional amount is calculated by currency.

Output

The output shows the total exposure by currency (sum of trade notional amounts) for single name credit default swap, credit index, and credit basket by party.

…
<singleNameCDSExposureByParty>
  <partyId>ABCBICXXX</partyId>
  <singleNameCDSExposureByCurrency>
    <currency>JPY</currency>
    <aggregateNotional>1.5E9</aggregateNotional>
  </singleNameCDSExposureByCurrency>
  <singleNameCDSExposureByCurrency>
    <currency>USD</currency>
    <aggregateNotional>2.068E8</aggregateNotional>
  </singleNameCDSExposureByCurrency>
  <singleNameCDSExposureByCurrency>
    <currency>EUR</currency>
    <aggregateNotional>2.0E7</aggregateNotional>

```
   </singleNameCDSExposureByCurrency>
</singleNameCDSExposureByParty>
…
<creditIndexExposureByParty>
   <partyId>PARTYABICXXX</partyId>
   <creditIndexExposureByCurrency>
      <currency>EUR</currency>
      <aggregateNotional>2.5E7</aggregateNotional>
   </creditIndexExposureByCurrency>
</creditIndexExposureByParty>…
<creditBasketExposureByParty>
   <partyId>MSCSBIC</partyId>
   <creditBasketExposureByCurrency>
      <currency>USD</currency>
      <aggregateNotional>7.5E7</aggregateNotional>
   </creditBasketExposureByCurrency>
   <creditBasketExposureByCurrency>
      <currency>EUR</currency>
      <aggregateNotional>1.0E7</aggregateNotional>
   </creditBasketExposureByCurrency>
</creditBasketExposureByParty>
…
```

Conclusions

Changes in regulatory requirements will lead in the near future to the reporting of all OTC derivative transactions to data repositories with full access for the regulator. In certain asset classes such as credit derivatives this reporting is happening already. For this increased reporting to be useful it is crucial that the means are available to analyze the data.

FpML is an XML data format that is widely used to represent OTC derivatives transactions and their processing. It is used in message exchanges and for storing the derivatives information.

The exercise we undertake demonstrates that there are multiple existing mechanisms freely available and widely used across multiple industries, such as XQuery, which allow extracting the data from an XML standard such as FpML to generate meaningful risk measures. The FpML data representation, were elements are tied into the ISDA legal definitions can be queried easily and provide the necessary level of detail to solve the questions at hand.

Some of the developed queries would require certain enhancements for an implementation in a real life environment, but each of them is able to retrieve the relevant data in order to answer the nine questions defined by the Canadian Securities Administrators' paper. The need for enhancements would not be solved by the presence of an additional layer of semantics but are related to FpML specificities such as schemes, references, and multiple trade, party, and entity identifiers.

Recent developments in the industry such as Legal Entity Identifier (LEI) and the Unique Product Identifier (UPI) will further facilitate the implementation of similar queries. Costly access to external reference data systems may be avoided if LEI is used as identification mechanism for entities and the product categorization code implemented in the queries could be simplified once taxonomies and UPI are part of the trade reporting.

FpML is closely working with industry and regulators to define standard messages for position and activity reporting. In this implementation we have not used these message formats to output the data but they could potentially be integrated in further refinements of the implementation.

Annex A: Implementation of the Regulatory Questions in XQuery

Question 1: Aggregate notional data for all contracts traded or settled in Canadian dollars, including a breakdown by reference entity and/or sector

```
declare default element namespace "http://www.fpml.org/FpML-5/reporting";

let $trade := collection()//trade[creditDefaultSwap/protectionTerms/calculationAmount/currency
= 'USD' or creditDefaultSwap/cashSettlementTerms/settlementCurrency= 'USD']

for $entityId in distinct-
values($trade/creditDefaultSwap/generalTerms/referenceInformation/referenceEntity/entityId)
let $byEntityId :=
$trade/creditDefaultSwap/generalTerms/referenceInformation/referenceEntity[entityId=$entityId]
return
<creditEntityExposure>
	<entityId>{$entityId}</entityId>
	<aggregateNotional>{sum($byEntityId/../../../protectionTerms/calculationAmount/amount
)}</aggregateNotional>
</creditEntityExposure>,

let $trade := collection()//trade[creditDefaultSwap/protectionTerms/calculationAmount/currency
= 'USD' or creditDefaultSwap/cashSettlementTerms/settlementCurrency= 'USD']
for $indexName in distinct-
values($trade/creditDefaultSwap/generalTerms/indexReferenceInformation/indexName)
let $byIndexName :=
$trade/creditDefaultSwap/generalTerms/indexReferenceInformation[indexName=$indexName]
return
<creditIndexExposure>
	<indexName>{$indexName}</indexName>
	<aggregateNotional>{sum($byIndexName/../../protectionTerms/calculationAmount/amo
unt)}</aggregateNotional>
</creditIndexExposure>,

let $trade := collection()//trade[returnSwap/returnLeg/notional/notionalAmount/currency = 'USD'
or returnSwap/returnLeg/amount/currency = 'USD']
for $instrumentId in distinct-
values($trade/returnSwap/returnLeg/underlyer/singleUnderlyer/*/instrumentId)
let $byInstrumentId :=
$trade/returnSwap/returnLeg/underlyer/singleUnderlyer/*[instrumentId=$instrumentId]
return
<equityExposure>
	<instrumentId>{$instrumentId}</instrumentId>
	<aggregateNotional>{sum($byInstrumentId/../../../notional/notionalAmount/amount)}</ag
gregateNotional>
</equityExposure>,

let $trade := collection()//trade[genericProduct/notional/currency = 'USD' or
genericProduct/settlementCurrency = ‘USD’]
for $instrumentId in distinct-values($trade/genericProduct/underlyer/*/instrumentId)
let $byInstrumentId :=
$trade/returnSwap/genericProduct/underlyer/*[instrumentId=$instrumentId]
return
<bespokeExposure>
	<instrumentId>{$instrumentId}</instrumentId>
	<aggregateNotional>{sum($byInstrumentId/../notional/amount)}</aggregateNotional>
</bespokeExposure>
```

Question 2: A list of all parties trading in USD with each party's aggregate notional position by product type and total notional amount

```
declare default element namespace "http://www.fpml.org/FpML-5/confirmation";
```

The query creates collections for USD trades for each product type. This is critical in order to calculate the aggregate notional by product type

```
let $cdTrade :=
collection()//trade[creditDefaultSwap/protectionTerms/calculationAmount/currency = 'USD']
let $returnSwapTrade :=
collection()//trade[returnSwap/returnLeg/notional/notionalAmount/currency = 'USD']
let $ratesTrade :=
collection()//trade[swap/swapStream/calculationPeriodAmount/calculation/notionalSchedule/notionalStepSchedule/currency = 'USD']
```

The trade collection is just the union of the other product trade collections:

```
let $trade := ($cdTrade, $returnSwapTrade, $ratesTrade)
```

Then it looks for all unique parties and it creates a loop to group the entities with the same id:

```
for $partyId in distinct-values($trade/../party/partyId)
let $byPartyId := $trade/../party[partyId=$partyId]
```

For all trades with the same party id, the sum function aggregates their notional amount for each product type.

```
let $creditExposure :=
sum($byPartyId/../trade/creditDefaultSwap/protectionTerms/calculationAmount/amount)
let $returnSwapExposure :=
sum($byPartyId/../trade/returnSwap/returnLeg/notional/notionalAmount/amount)
let $interestRateSwapExposure :=
sum($byPartyId/../trade/swap/swapStream/calculationPeriodAmount/calculation/notionalSchedule/notionalStepSchedule/initialValue)
```

It aggregates the total notional including all products by party id.

```
let $totalExposure := $creditExposure + $returnSwapExposure + $interestRateSwapExposure
```

The party aggregates are ordered by total exposure descending:

```
order by $totalExposure descending

return
<totalExposureByParty>
        <partyId>{$partyId}</partyId>
        <creditExposure>{$creditExposure}</creditExposure>
        <returnSwapExposure>{$returnSwapExposure}</returnSwapExposure>
        <interestRateSwapExposure>{$interestRateSwapExposure}</interestRateSwapExposure>
        <totalExposure>{$totalExposure}</totalExposure>
</totalExposureByParty>
```

Question 3: A list of the top counterparty positions for each of the largest financial groups in Canada

```
declare default element namespace "http://www.fpml.org/FpML-5/confirmation";
```

```
let $trade := collection()//trade[exists(creditDefaultSwap/generalTerms/referenceInformation or
creditDefaultSwap/generalTerms/indexReferenceInformation)]
for $partyId in distinct-values($trade/../party/partyId)
let $byPartyId := $trade/../party[partyId=$partyId]
return
<creditPositionsByParty>
        <partyId>{$partyId}</partyId>{
        for $partyTrade in $byPartyId/../trade
        let $tradeId := $partyTrade//tradeHeader/partyTradeIdentifier[1]/tradeId
        let $notionalCurrency :=
$partyTrade/creditDefaultSwap/protectionTerms/calculationAmount/currency
        let $notionalAmount :=
$partyTrade/creditDefaultSwap/protectionTerms/calculationAmount/amount
        order by $notionalAmount descending
        return
        <position>
                {$tradeId}
                <notional>
                        {$notionalCurrency}
                        {$notionalAmount}
                </notional>
        </position>}
</creditPositionsByParty>,

let $trade := collection()//trade[exists(returnSwap/returnLeg)]
for $partyId in distinct-values($trade/../party/partyId)
let $byPartyId := $trade/../party[partyId=$partyId]
return
<returnSwapPositionsByParty>
        <partyId>{$partyId}</partyId>{
        for $partyTrade in $byPartyId/../trade
        let $tradeId := $partyTrade/tradeHeader/partyTradeIdentifier[1]/tradeId
        let $notionalCurrency :=
$partyTrade/returnSwap/returnLeg/notional/notionalAmount/currency
        let $notionalAmount :=
$partyTrade/returnSwap/returnLeg/notional/notionalAmount/amount
        order by $notionalAmount descending
        return
        <position>
                {$tradeId}
                <notional>
                        {$notionalCurrency}
                        {$notionalAmount}
                </notional>
        </position>}
</returnSwapPositionsByParty>,

let $trade := collection()//trade[exists(swap/swapStream)]
for $partyId in distinct-values($trade/../party/partyId)
let $byPartyId := $trade/../party[partyId=$partyId]
return
<interestRateSwapPositionsByParty>
        <partyId>{$partyId}</partyId>{
        for $partyTrade in $byPartyId/../trade
        let $tradeId := $partyTrade/tradeHeader/partyTradeIdentifier[1]/tradeId
        let $notionalCurrency :=
$partyTrade/swap/swapStream[1]/calculationPeriodAmount/calculation/notionalSchedule/notion
alStepSchedule/currency
```

```
	let $notionalAmount :=
$partyTrade/swap/swapStream[1]/calculationPeriodAmount/calculation/notionalSchedule/notion
alStepSchedule/initialValue
	order by $notionalAmount descending
	return
	<position>
		{$tradeId}
		<notional>
			{$notionalCurrency}
			{$notionalAmount}
		</notional>
	</position>}
</interestRateSwapPositionsByParty>
```

Question 4: Aggregate notional data for contracts written on Canadian domiciled corporations (reference entities), including a list of the top aggregate notional counterparty positions for contracts written on each firm

```
declare default element namespace "http://www.fpml.org/FpML-5/confirmation";

let $trade :=
collection()//trade[exists(creditDefaultSwap/generalTerms/referenceInformation/referenceEntity)]
for $entityId in distinct-
values($trade/creditDefaultSwap/generalTerms/referenceInformation/referenceEntity/entityId)
let $byEntityId :=
$trade/creditDefaultSwap/generalTerms/referenceInformation/referenceEntity[entityId=$entityId]
return
<creditPositionsByEntityId>
	<entityId>{$entityId}</entityId>{
	for $currency in distinct-
values($byEntityId/../../../../creditDefaultSwap/protectionTerms/calculationAmount/currency)
	let $byCurrency :=
$byEntityId/../../../../creditDefaultSwap/protectionTerms/calculationAmount[currency=$currency]
	let $aggregateNotional := sum($byCurrency/amount)
	return
	<creditPositionsByCurrency>
		<aggregateNotional>
			<currency>{$currency }</currency>
			<amount>{$aggregateNotional}</amount>
		</aggregateNotional>{
		for $currencyTrade in $byCurrency/../../../../trade
		let $tradeId := $currencyTrade/tradeHeader/partyTradeIdentifier[1]/tradeId
		let $notionalAmount :=
$currencyTrade/creditDefaultSwap/protectionTerms/calculationAmount/amount
		order by $notionalAmount descending
		return
		<position>
			{$tradeId}
			<notional>
				{$notionalAmount}
			</notional>
		</position>}
	</creditPositionsByCurrency>}
</creditPositionsByEntityId>
```

Question 5: A list of the top counterparties' aggregate notional positions where the contract references the debt of the government of Canada

```
declare default element namespace "http://www.fpml.org/FpML-5/confirmation";

let $trade :=
collection()//trade[creditDefaultSwap/generalTerms/referenceInformation/referenceObligation/bo
nd/issuer = 'Her Majesty in right of Canada' or
creditDefaultSwap/generalTerms/referenceInformation/referenceEntity/entityId = '27CBJG' or
creditDefaultSwap/generalTerms/referenceInformation/referenceEntity/entityName = 'Canada']

for $partyId in distinct-values($trade/../party/partyId)
let $byPartyId := $trade/../party[partyId=$partyId]
return
<creditExposureByParty>
	<partyId>{$partyId}</partyId>{
	for $currency in distinct-
values($byPartyId/../trade/creditDefaultSwap/protectionTerms/calculationAmount/currency)
	let $byCurrency :=
$byPartyId/../trade/creditDefaultSwap/protectionTerms/calculationAmount[currency=$currency]
	let $aggregateNotional := sum($byCurrency/amount)
	return
	<creditExposureByCurrency>
		<aggregateNotional>
			<currency>{$currency }</currency>
			<amount>{$aggregateNotional}</amount>
		</aggregateNotional>
	</creditExposureByCurrency>}
</creditExposureByParty>
```

Question 6: A list of the top counterparties' aggregate notional positions where the contract references a specific commodity

```
declare default element namespace "http://www.fpml.org/FpML-5/confirmation";

let $trade := collection()//trade[contains(commoditySwap//commodity/instrumentId,
'ELECTRICITY')]
for $partyId in distinct-values($trade/../party/partyId)
let $byPartyId := $trade/../party[partyId=$partyId]
return
<commodityExposureByParty>
	<partyId>{$partyId}</partyId>
	<powerExposure>{sum($byPartyId/../trade//totalNotionalQuantity)}</powerExposure>
</commodityExposureByParty>,

let $trade := collection()//trade[contains(commoditySwap//commodity/instrumentId, 'GAS')]
for $partyId in distinct-values($trade/../party/partyId)
let $byPartyId := $trade/../party[partyId=$partyId]
return
<commodityExposureByParty>
	<partyId>{$partyId}</partyId>
	<gasExposure>{sum($byPartyId/../trade//totalNotionalQuantity)}</gasExposure>
</commodityExposureByParty>,

let $trade := collection()//trade[contains(commoditySwap//commodity/instrumentId, 'COAL')]
for $partyId in distinct-values($trade/../party/partyId)
```

```
let $byPartyId := $trade/../party[partyId=$partyId]
return
<commodityExposureByParty>
        <partyId>{$partyId}</partyId>
        <coalExposure>{sum($byPartyId/../trade//totalNotionalQuantity)}</coalExposure>
</commodityExposureByParty>,

let $trade := collection()//trade[contains(commoditySwap//commodity/instrumentId, 'OIL')]
for $partyId in distinct-values($trade/../party/partyId)
let $byPartyId := $trade/../party[partyId=$partyId]
return
<commodityExposureByParty>
        <partyId>{$partyId}</partyId>
        <oilExposure>{sum($byPartyId/../trade//totalNotionalQuantity)}</oilExposure>
</commodityExposureByParty>
```

Question 7: A list of the top counterparties' aggregate notional positions where the contracts reference the debt of one of the ten largest Canadian financial groups

```
declare default element namespace "http://www.fpml.org/FpML-5/confirmation";

let $trade :=
collection()//trade[creditDefaultSwap/generalTerms/referenceInformation/referenceObligation/bo
nd/instrumentId = 'JP310504B117'] (:database lookup would need to be performed to retrieve
ISIN/CUSIP from Canadian corporations:)

for $partyId in distinct-values($trade/../party/partyId)
let $byPartyId := $trade/../party[partyId=$partyId]
return
<creditExposureByParty>
        <partyId>{$partyId}</partyId>{
        for $currency in distinct-
values($byPartyId/../trade/creditDefaultSwap/protectionTerms/calculationAmount/currency)
        let $byCurrency :=
$byPartyId/../trade/creditDefaultSwap/protectionTerms/calculationAmount[currency=$currency]
        let $aggregateNotional := sum($byCurrency/amount)
        return
        <creditExposureByCurrency>
                <aggregateNotional>
                        <currency>{$currency }</currency>
                        <amount>{$aggregateNotional}</amount>
                </aggregateNotional>
        </creditExposureByCurrency>}
<creditExposureByParty>
```

Question 8: Data on the overall level of activity of each of the Canadian banks in each asset class

```
declare default element namespace "http://www.fpml.org/FpML-5/confirmation";

let $cdTrade := collection()//trade[exists(creditDefaultSwap/generalTerms/referenceInformation
or creditDefaultSwap/generalTerms/indexReferenceInformation)]
let $returnSwapTrade := collection()//trade[exists(returnSwap or
equitySwapTransactionSupplement)]
```

```
let $ratesTrade := collection()//trade[exists(swap)]
let $trade := ($cdTrade, $returnSwapTrade, $ratesTrade)
for $partyId in distinct-values($trade/../party/partyId) (:database lookup would be needed to look
for specific Canadian banks :)
let $byPartyId := $trade/../party[partyId=$partyId]

return
<totalExposureByParty>
        <partyId>{$partyId}</partyId>{
        for $currency1 in distinct-
values($byPartyId/../trade/creditDefaultSwap/protectionTerms/calculationAmount/currency)
        let $byCurrency1 :=
$byPartyId/../trade/creditDefaultSwap/protectionTerms/calculationAmount[currency=$currency1]
        let $creditExposure := sum($byCurrency1/amount)
        return
        <creditExposureByCurrency>
                <currency>{$currency1}</currency>
                <aggregateNotional>{$creditExposure}</aggregateNotional>
        </creditExposureByCurrency>}{
        for $currency2 in distinct-
values($byPartyId/../trade/returnSwap/returnLeg/notional/notionalAmount/amount)
        let $byCurrency2 :=
$byPartyId/../trade/returnSwap/returnLeg/notional/notionalAmount[currency=$currency2]
        let $equityExposure := sum($byCurrency2/amount)
        return
        <equityExposureByCurrency>
                <currency>{$currency2}</currency>
                <aggregateNotional>{$equityExposure}</aggregateNotional>
        </equityExposureByCurrency>}{
        for $currency3 in distinct-
values($byPartyId/../trade/swap/swapStream/calculationPeriodAmount/calculation/notionalSche
dule/notionalStepSchedule/currency)
        let $byCurrency3 :=
$byPartyId/../trade/swap/swapStream/calculationPeriodAmount/calculation/notionalSchedule/no
tionalStepSchedule[currency=$currency3]
        let $ratesExposure := sum($byCurrency3/initialValue)
        return
        <ratesExposureByCurrency>
                <currency>{$currency3}</currency>
                <aggregateNotional>{$ratesExposure}</aggregateNotional>
        </ratesExposureByCurrency>}</totalExposureByParty>
```

Question 9: Each of the Canadian bank's overall positions in specific products within an asset class

```
declare default element namespace "http://www.fpml.org/FpML-5/confirmation";

let $cdsTrade :=
collection()//trade[exists(creditDefaultSwap/generalTerms/referenceInformation)]
for $partyId in distinct-values($cdsTrade/../party/partyId) (:database lookup would be needed to
look for specific Canadian banks :)
let $byPartyId := $cdsTrade/../party[partyId=$partyId]
return
<singleNameCDSExposureByParty>
        <partyId>{$partyId}</partyId>{
        for $currency in distinct-
values($byPartyId/../trade/creditDefaultSwap/protectionTerms/calculationAmount/currency)
```

```
		let $byCurrency :=
$byPartyId/../trade/creditDefaultSwap/protectionTerms/calculationAmount[currency=$currency]
		let $aggregateNotional := sum($byCurrency/amount)
		return
		<singleNameCDSExposureByCurrency>
			<currency>{$currency}</currency>
			<aggregateNotional>{$aggregateNotional}</aggregateNotional>
		</singleNameCDSExposureByCurrency>}
</singleNameCDSExposureByParty>,

let $cdIndexTrade :=
collection()//trade[exists(creditDefaultSwap/generalTerms/indexReferenceInformation)]
for $partyId in distinct-values($cdIndexTrade/../party/partyId) (:database lookup would be
needed to look for specific Canadian banks :)
let $byPartyId := $cdIndexTrade/../party[partyId=$partyId]
return
<creditIndexExposureByParty>
		<partyId>{$partyId}</partyId>{
		for $currency in distinct-
values($byPartyId/../trade/creditDefaultSwap/protectionTerms/calculationAmount/currency)
		let $byCurrency :=
$byPartyId/../trade/creditDefaultSwap/protectionTerms/calculationAmount[currency=$currency]
		let $aggregateNotional := sum($byCurrency/amount)
		return
		<creditIndexExposureByCurrency>
			<currency>{$currency}</currency>
			<aggregateNotional>{$aggregateNotional}</aggregateNotional>
		</creditIndexExposureByCurrency>}
</creditIndexExposureByParty>,

let $cdsBasketTrade :=
collection()//trade[exists(creditDefaultSwap/generalTerms/basketReferenceInformation)]
for $partyId in distinct-values($cdsBasketTrade/../party/partyId) (:database lookup would be
needed to look for specific Canadian banks :)
let $byPartyId := $cdsBasketTrade/../party[partyId=$partyId]
return
<creditBasketExposureByParty>
		<partyId>{$partyId}</partyId>{
		for $currency in distinct-
values($byPartyId/../trade/creditDefaultSwap/protectionTerms/calculationAmount/currency)
		let $byCurrency :=
$byPartyId/../trade/creditDefaultSwap/protectionTerms/calculationAmount[currency=$currency]
		let $aggregateNotional := sum($byCurrency/amount)
		return
		<creditBasketExposureByCurrency>
			<currency>{$currency}</currency>
			<aggregateNotional>{$aggregateNotional}</aggregateNotional>
		</creditBasketExposureByCurrency>}
</creditBasketExposureByParty>,

let $cdOption := collection()//trade[exists(creditDefaultSwapOption)]
for $partyId in distinct-values($cdOption/../party/partyId) (:database lookup would be needed to
look for specific Canadian banks :)
let $byPartyId := $cdOption/../party[partyId=$partyId]
return
<creditOptionExposureByParty>
		<partyId>{$partyId}</partyId>{
		for $currency in distinct-
values($byPartyId/../trade//creditDefaultSwap/protectionTerms/calculationAmount/currency)
```

```
        let $byCurrency :=
$byPartyId/../trade//creditDefaultSwap/protectionTerms/calculationAmount[currency=$currency]
        let $aggregateNotional := sum($byCurrency/amount)
        return
        <creditOptionExposureByCurrency>
                <currency>{$currency}</currency>
                <aggregateNotional>{$aggregateNotional}</aggregateNotional>
        </creditOptionExposureByCurrency>}
</creditOptionExposureByParty>
```