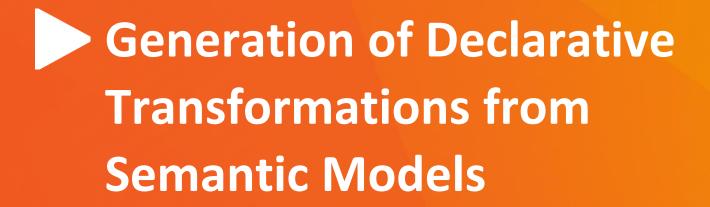
Vladimir Alexiev, PhD, PMP

Ontotext Corp, Sofia, Bulgaria





#### Outline

- Building Knowledge Graphs in 10 steps
- Semantic Modeling vs Ontology Engineering; Polyglot Modeling
- rdfpuml: semantic models from turtle examples
- rdf2sparql: tabular transforms from turtle examples
- rdf2rml: R2RML transforms from turtle examples
- Backup slides: Semantic ETL desiderata
- Next time: tsv2owl, tsv2soml, owl2soml: ontologies and Semantic Object models from tabular data





#### **Building Knowledge Graphs in 10 steps**





2 Gather and analyze relevant data



3 Clean data to ensure data quality



4 Create your semantic data model



5 Integrate data with ETL or virtualization



Harmonize data via reconciliation, fusion and alignment



Architect the data management and search layer



Augment your graph via reasoning analytics and text analysis



Maximize the usability of your data



Make your KG easy to maintain and evlove

- I'm Chief Data Architect at Ontotext, working at the Knowledge Graph Solutions group. We build KGs for a living
- Above is a simple <u>10-step recipe for KG building</u>
  - Webinar: Knowledge Graphs: 5 Use Cases and 10 Steps to Get There
  - Video: <u>Building Knowledge Graphs in 10 Steps</u>
- This presentation is about steps 4 (models) and 5 (semantic ETL)



# **Semantic Modeling**



# Semantic Modeling vs Ontology Engineering

- Ontology engineering:
  - Know methodologies, modeling principles, top-level ontologies
  - Create ontologies mostly from scratch (green-field)
- Semantic modeling:
  - Research existing data standards and assets, harmonize them
  - Know and reuse existing ontologies, understand tradeoffs
  - Add classes and props when needed
  - Make holistic models that show how ontologies are used together
  - Document (human readable) and make machine readable artifacts (application profiles, RDF shapes, APIs, etc)
- For me semantic modeling is more than ontologies
  - Also, we like to do it in a very pragmatic way



# Polyglot Modeling

- Simple, technology-independent models
  - Understandable by domain experts: increases participation
- Generate various tech artefacts: semantic web doesn't (yet) rule the world!
  - Diagrams, documentation
  - RDFS, OWL, SHACL, SHEX, JSON-LD Context and Frame,
  - JSON schema, Elastic object model, etc, etc
  - "Write once. Use many. Creative laziness encouraged" [RAML]

#### Examples:

- FHIR: domain model to XML, JSON, JSONLD; XSD, JSON Schema, SHEX
- <u>LinkML</u>: YAML-based models to various technical artefacts incl. JSON Schema, ontology, SHEX
- Ontotext SOML: YAML-based model to SHACL and GraphQL (exposes KG parts to GraphQL)
- Schema Salad (part of CWL): YAML-based model to JSON Schema and RDFS
- A.ML and CloudInformationModel: YAML to RDFS, SHACL, SQL, R2RML
- RAML (RESTful API Modeling Language): YAML to APIs (ala OpenAPI specifications)

#### Links:

- Many of these are in YAML: YAML-LD work group started (part of JSON-LD CG), yaml-ld#19 is about polyglot modeling
- JSON-LD, YAML-LD and Polyglot Modeling. Alexiev, V. presentation, October 2022. <u>slides link</u>
- Data Spaces vs Knowledge Graphs: How to Get To Semantic Data Spaces? Alexiev, V. In Data Spaces & Semantic Interoperability Workshop, Vienna, Austria, July 2022. Paper slides blog bibbase



# Semantic Models from Turtle Examples

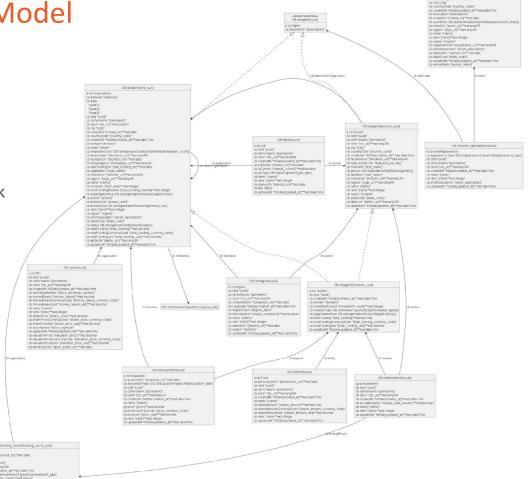
- https://github.com/VladimirAlexiev/rdf2rml: several open source tools.
   The first tool is:
- rdfpuml: generates model diagrams from examples in turtle. Lots of information and precision:
  - Ontologies, specific classes and properties
  - URL patterns
  - Datatypes and language tags
  - Simple data cleaning (e.g. SPLIT, URLIFY, FIX\_DATE)
  - Use of source data (in parentheses)
  - Combining fields into the same property (e.g. alias1,2,3 mapped to multi-valued cb:alias)
  - Data of some nodes comes from multiple tables
- This is an actionable model, no hand-waving here
  - RDF by Example: rdfpuml for True RDF Diagrams, rdf2rml for R2RML Generation. Alexiev, V. Semantic Web in Libraries 2016 (SWIB 2016), Bonn, Germany, November 2016. <u>slides</u> <u>html</u> <u>video</u> <u>bibbase</u>



#### **Crunchbase Semantic Model**

<u>Crunchbase</u> is a popular company dataset

- 18 CSV tables (with uuid and updated\_at), 10M records, updated daily
- 2M companies, 929k detailed descriptions, 22k org parent links, 429k event appearances
- 1.5M people, 848k descriptions, 2.2M job positions, 495k degrees (education records)
- 223k investors, 488k funding rounds, 783k investments (participation in funding round), 137k acquisitions, 48k IPOs
- 750 industry classifications
- (Full image in slightly different style)





,∕cb:organization

#### Simple custom ontology:

- Exchange (exchangeSymbol) and ticker (stockSymbol) (e.g. <u>NASDAQ:MSFT</u>)
- Who (organization), when (wentPublicOn)
- Financial: moneyRaised, sharePrice, valuationPrice (in local currency and USD)
- cbPermalink, cbUrl (e.g. <u>Microsoft IPO</u>), CB rank
- Bookkeeping: uuid, createdAt, updatedAt (allow daily update)

#### Model elements:

- URL patterns
- Datatypes (e.g. ^^xsd:integer)
- Simple fixes (e.g. **fixDate**)

#### CB:cb/ipo/(uuid)

```
a cb:IPO
cb:cbld "[uuid]"
cb:cbPermalink "[permalink]"
cb:cbUrl "[cb_url]"^xsd:anvURI
cb:createdAt "fixDate[created at]"^^xsd:dateTime
cb:exchangeSymbol "[stock exchange symbol]"
cb:moneyRaised "[money raised]"^xsd:decimal
cb:moneyRaisedCurrencyCode "[money_raised_currency_code]"
cb:moneyRaisedUsd "[money raised usd]"^xsd:decimal
cb:name "[name]"
cb:rank "[rank]"^xsd:integer
cb:sharePrice "[share price]"^xsd:decimal
cb:sharePriceCurrencyCode "[share_price_currency_code]"
cb:sharePriceUsd "[share price usd]"^xsd:decimal
cb:stockSymbol "[stock symbol]"
cb:updatedAt "fixDate[updated_at]"^xsd:dateTime
cb:valuationPrice "[valuation_price]"^xsd:decimal
cb:valuationPriceCurrencyCode "[valuation price currency code]"
cb:valuationPriceUsd "[valuation_price_usd]"^xsd:decimal
cb:wentPublicOn "[went_public_on]"^^xsd:date
```

# **Generating Tabular Transforms**



# **Tabular Transforms From Turtle Examples**

- rdf2sparql: generates SPARQL queries for transforming tabular data
- UPDATE for Ontotext Refine
  - Update directly to semantic repository: faster
  - Can join tabular to RDF data in a repository.
  - Wamed-graph per table" for updates (locality and idempotence)
  - ⑤ For CB we used "named graph per row" for fast daily update: GraphDB handles 10M named graphs just fine
- CONSTRUCT for Ontotext Refine.
  - Generates intermediate RDF file: slower
  - Can join tabular to RDF data in a repository
- CONSTRUCT queries for <u>TARQL</u>.
  - Generates intermediate RDF file: slower
  - Cannot join tabular to RDF data in a repository
- Handles datatypes, templated URLs, simple data cleaning (functions/macros)
- Chains functions applied to the same field (see example on the right)

- Turtle model using 2 functions:
   cb:employeeCount <cb/employeeCount/urlify(ifNotNull(employee count))>;
- Macros that define the functions:
   #define urlify(x) bind(LCASE(REPLACE(REPLACE(REPLACE
   (x, "[^\\p{L}0-9]", "\_"), "\_+", "\_"), "^\_|\_\$", "")) as x##\_URLIFY)
   #define ifNotNull(x) bind(if(x in
   ("other","not provided","unknown"),?UNDEF,x) as x##\_IFNOTNULL)
- SPARQL CONSTRUCT/UPDATE:
   cb:employeeCount
   ?cb\_employeeCount\_employee\_count\_IFNOTNULL\_URLIFY\_URL;

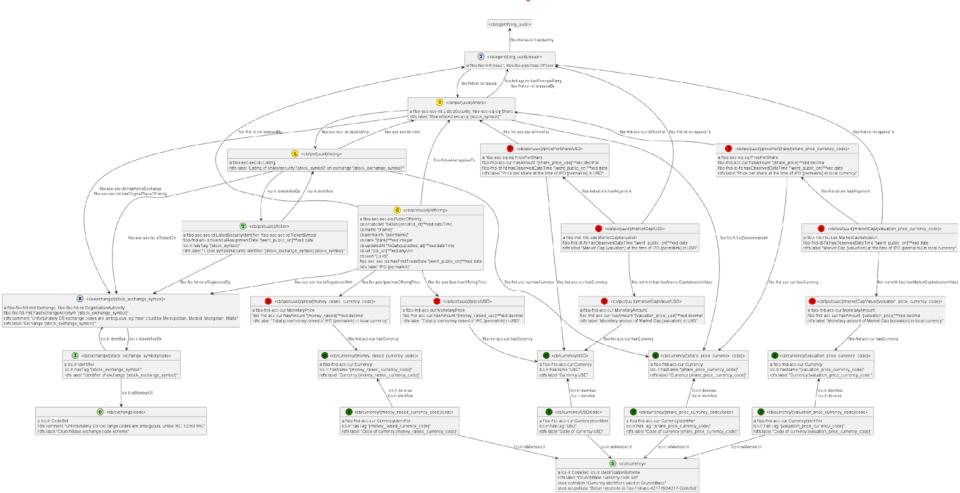
#### CB IPOs: From Model to SPARQL UPDATE

```
<cb/ipo/(uuid)> a cb:IPO:
 cb:cbId '(uuid)':
 cb:name '(name)';
 cb:cbPermalink '(permalink)';
 cb:cbUrl '(cb url)'^^xsd:anyURI;
 cb:rank '(rank)'^^xsd:integer;
 cb:createdAt 'fixDate(created at)'^^xsd:dateTime;
 cb:updatedAt 'fixDate(updated at)'^^xsd:dateTime;
 cb:organization <cb/agent/(org_uuid)>;
 cb:exchangeSymbol '(stock_exchange_symbol)';
 cb:stockSymbol '(stock_symbol)';
 cb:wentPublicOn '(went_public_on)'^^xsd:date;
 cb:sharePriceUsd '(share price usd)'^^xsd:decimal;
 cb:sharePrice '(share_price)'^^xsd:decimal;
 cb:sharePriceCurrencyCode '(share_price_currency_code)';
 cb:valuationPriceUsd '(valuation_price_usd)'^^xsd:decimal;
 cb:valuationPrice '(valuation price)'^^xsd:decimal;
 cb:valuationPriceCurrencyCode '(valuation price currency code)';
 cb:moneyRaisedUsd '(money_raised_usd)'^^xsd:decimal;
 cb:monevRaised '(monev raised)'^^xsd:decimal:
 cb:moneyRaisedCurrencyCode '(money_raised_currency_code)'.
```

```
delete {graph ?GRAPH {?s ?p ?o}}
where {
  service <rdf-mapper:ontorefine:PROJECT_ID> {
    bind(?c updated at as ?c updated at1)
    bind(iri(concat("cb/graph/ipos/",?c uuid)) as ?GRAPH)}
  <cb> cb:updatedAt ?UPDATED AT DATETIME.
  bind(replace(str(?UPDATED_AT_DATETIME),'T',' ') as ?UPDATED_AT)
  filter(?c updated at1 > ?UPDATED AT)
  graph ?GRAPH {?s ?p ?o}};
insert {graph ?GRAPH {
  ?cb ipo uuid URL a cb:IPO;
    cb:cbId ?uuid:
    cb:name ?name;
    cb:cbPermalink ?permalink;
    cb:cbUrl ?CB URL;
    cb:rank ?RANK;
    cb:createdAt ?CREATED AT FIXDATE;
    cb:updatedAt ?UPDATED AT FIXDATE;
    cb:organization ?cb agent org uuid URL;
    cb:exchangeSymbol ?stock exchange symbol:
    cb:stockSymbol ?stock symbol;
    cb:wentPublicOn ?WENT PUBLIC ON;
    cb:sharePriceUsd ?SHARE PRICE USD;
    cb:sharePrice ?SHARE PRICE;
    cb:sharePriceCurrencyCode ?share price currency code;
    cb:valuationPriceUsd ?VALUATION PRICE USD;
    cb:valuationPrice ?VALUATION PRICE:
    cb:valuationPriceCurrencyCode ?valuation price currency code;
    cb:monevRaisedUsd ?MONEY RAISED USD:
    cb:moneyRaised ?MONEY RAISED;
    cb:moneyRaisedCurrencyCode ?money raised currency code.
where {
```

```
service <rdf-mapper:ontorefine:PROJECT ID> {
  bind(?c uuid as ?uuid)
  bind(?c name as ?name)
  bind(?c permalink as ?permalink)
  bind(?c cb url as ?cb url)
  bind(?c rank as ?rank)
  bind(?c created at as ?created at)
  bind(?c updated at as ?updated at)
  bind(?c org uuid as ?org uuid)
  bind(?c_stock_exchange_symbol as ?stock_exchange_symbol)
  bind(?c stock symbol as ?stock symbol)
  bind(?c went public on as ?went public on)
  bind(?c share price usd as ?share price usd)
  bind(?c share price as ?share price)
  bind(?c share price currency code as ?share price currency code)
  bind(?c valuation price usd as ?valuation price usd)
  bind(?c_valuation_price as ?valuation_price)
  bind(?c_valuation_price_currency_code as ?valuation_price_currency_code)
  bind(?c_money_raised_usd as ?money_raised_usd)
  bind(?c money raised as ?money raised)
  bind(?c money raised currency code as ?money raised currency code)
  bind(iri(concat("cb/ipo/",?uuid)) as ?cb ipo uuid URL)
  bind(strdt(?cb url,xsd:anyURI) as ?CB URL)
  bind(strdt(?rank,xsd:integer) as ?RANK)
  bind(REPLACE(?created_at,' ','T') as ?created_at_FIXDATE)
  bind(strdt(?created_at_FIXDATE,xsd:dateTime) as ?CREATED_AT_FIXDATE)
  bind(REPLACE(?updated_at,' ','T') as ?updated_at_FIXDATE)
  bind(strdt(?updated at FIXDATE.xsd:dateTime) as ?UPDATED AT FIXDATE)
  bind(iri(concat("cb/agent/",?org uuid)) as ?cb agent org uuid URL)
  bind(strdt(?went public on,xsd:date) as ?WENT PUBLIC ON)
  bind(strdt(?share price usd,xsd:decimal) as ?SHARE PRICE USD)
  bind(strdt(?share_price,xsd:decimal) as ?SHARE_PRICE)
  bind(strdt(?valuation price usd,xsd:decimal) as ?VALUATION PRICE USD)
  bind(strdt(?valuation price,xsd:decimal) as ?VALUATION PRICE)
  bind(strdt(?money_raised_usd,xsd:decimal) as ?MONEY_RAISED_USD)
  bind(strdt(?money raised.xsd:decimal) as ?MONEY RAISED)
  bind(?c updated at as ?c updated at1)
 bind(iri(concat("cb/graph/ipos/",?c uuid)) as ?GRAPH)}
<cb> cb:updatedAt ?UPDATED AT DATETIME.
bind(replace(str(?UPDATED AT DATETIME), 'T', ' ') as ?UPDATED AT)
filter(?c updated at1 > ?UPDATED AT)}
```

# Could Go a Lot More Complex: IPOs in FIBO



# IPOs in FIBO: Turtle model (165 lines)

```
GRAPH <cb/ipos>
                                                                               lcc-lr:isIdentifiedBy <cb/ipo/(uuid)/ticker>;
                                                                               fibo-sec-sec-lst:isTradedOn <cb/exchange/(stock_exchange_symbol)>;
## Issuer as role of a CB agent
                                                                               fibo-fnd-acc-cur:hasCurrency <cb/currency/(share price currency code)>.
<cb/agent/(org uuid)/issuer> a fibo-fbc-fi-fi:Issuer, fibo-fbc-pas-fpas:Of-> <cb/ipo/(uuid)/ticker> a fibo-sec-sec-id:TickerSymbol, fibo-sec-sec-id:Lis
  fibo-fnd-rel-rel:issues <cb/ipo/(uuid)/share>;
                                                                               rdfs:label "Ticker symbol/security identifier '(stock_exchange_symbol):()
  fibo-fnd-rel-rel:hasIdentity <cb/agent/(org uuid)>.
                                                                               lcc-lr:hasTag '(stock symbol)';
                                                                               fibo-fbc-fct-ra:isRegisteredBy <cb/exchange/(stock exchange symbol)>;
                                                                               lcc-lr:identifies <cb/ipo/(uuid)/listing>;
## Stock exchange
                                                                               fibo-fnd-arr-id:hasInitialAssignmentDate '(went public on)'^^xsd:date.
<cb/exchange/(stock exchange symbol)> a fibo-fbc-fct-mkt:Exchange, fibo-fbd
                                                                                                                                                           1cc-1r:hasName 'USD';
  rdfs:label "Exchange '(stock exchange symbol)'";
                                                                             ## Financials
                                                                                                                                                           rdfs:label 'Currency USD'.
  rdfs:comment "Unfortunately CB exchange codes are ambiguous, eg 'mse' cou
  fibo-fbc-fct-mkt:hasExchangeAcronym '(stock exchange symbol)';
                                                                             <cb/ipo/(uuid)/price/(money raised currency code)> a fibo-fnd-acc-cur:Mone >
                                                                                                                                                           rdfs:label 'Code of currency USD':
                                                                               rdfs:label "Total price/money raised of IPO (permalink) in local currence
  lcc-lr:isIdentifiedBy <cb/exchange/(stock exchange symbol)/code>.
                                                                                                                                                           lcc-lr:hasTag 'USD';
                                                                               fibo-fnd-acc-cur:hasAmount '(money raised)'^^xsd:decimal;
                                                                                                                                                           lcc-lr:denotes <cb/currency/USD>;
<cb/exchange/(stock_exchange_symbol)/code> a lcc-lr:Identifier;
                                                                               fibo-fnd-acc-cur:hasCurrency <cb/currency/(money_raised_currency_code)>.
                                                                                                                                                           lcc-lr:identifies <cb/currency/USD>;
  rdfs:label "Identifier of exchange '(stock exchange symbol)'";
                                                                                                                                                           lcc-lr:isMemberOf <cb/currency>.
  lcc-lr:hasTag
                     '(stock exchange symbol)';
                                                                             <cb/ipo/(uuid)/price/USD> a fibo-fnd-acc-cur:MonetaryPrice:
  lcc-lr:identifies <cb/exchange/(stock exchange symbol)>;
                                                                               rdfs:label "Total price/money raised of IPO (permalink) in USD";
  lcc-lr:isMemberOf <cb/exchange/code>.
                                                                               fibo-fnd-acc-cur:hasAmount '(money raised usd)'^^xsd:decimal:
                                                                               fibo-fnd-acc-cur:hasCurrency <cb/currency/USD>.
<cb/exchange/code> a lcc-lr:CodeSet:
  rdfs:label 'CrunchBase exchange code scheme';
                                                                             <cb/ipo/(uuid)/marketCap/(valuation price currency code)> a fibo-ind-mkt-b>
  rdfs:comment 'Unfortunately CB Exchange codes are ambiguous, unlike ISO 3
                                                                               rdfs:label "Market Cap (valuation) at the time of IPO (permalink) in loc-
                                                                               fibo-ind-mkt-bas:hasMarketCapitalizationValue <cb/ipo/(uuid)/marketCapVa
## Offering, Share, Listing, Ticker
                                                                               fibo-fnd-utl-alx:hasArgument <cb/ipo/(uuid)/pricePerShare/(share_price_cr)
                                                                               fibo-fnd-dt-fd:hasObservedDateTime '(went public on)'^^xsd:date;
                                                                                                                                                           lcc-lr:isMemberOf <cb/currency>.
<cb/ipo/(uuid)/offering> a fibo-sec-sec-iss:PublicOffering;
                                                                               fibo-fnd-rel-rel:appliesTo <cb/agent/(org uuid)/issuer>.
  rdfs:label "IPO (permalink)";
  fibo-fbc-pas-fpas:hasOfferingPrice <cb/ipo/(uuid)/price/(money_raised_cui+ <cb/ipo/(uuid)/marketCap/USD> a fibo-ind-mkt-bas:MarketCapitalization;
  fibo-fnd-rel-rel:appliesTo <cb/ipo/(uuid)/share>;
                                                                               rdfs:label "Market Cap (valuation) at the time of IPO (permalink) in USD>
  fibo-fnd-rel-rel:isIssuedBy <cb/agent/(org uuid)/issuer>;
                                                                               fibo-ind-mkt-bas:hasMarketCapitalizationValue <cb/ipo/(uuid)/marketCapVa
  fibo-sec-sec-iss:hasFirstTradeDate '(went public on)'^^xsd:date;
                                                                               fibo-fnd-utl-alx:hasArgument <cb/ipo/(uuid)/pricePerShare/USD>;
  fibo-sec-sec-iss:isRegisteredWith <cb/exchange/(stock_exchange_symbol)>;
                                                                               fibo-fnd-dt-fd:hasObservedDateTime '(went_public_on)'^^xsd:date;
                                                                               fibo-fnd-rel-rel:appliesTo <cb/agent/(org_uuid)/issuer>.
  cb:uuid '(uuid)';
  cb:name '(name)';
  cb:permalink '(permalink)';
                                                                                                                                                           lcc-lr:isMemberOf <cb/currency>.
                                                                             <cb/ipo/(uuid)/marketCapValue/(valuation price currency code)> a fibo-fnd->
  cb:url '(cb_url)'^^xsd:anyURI;
                                                                               rdfs:label "Monetary amount of Market Cap (valuation) in local currency"
  cb:rank '(rank)'^^xsd:integer:
                                                                               fibo-fnd-acc-cur:hasAmount '(valuation price)'^^xsd:decimal:
  cb:createdAt 'fixDate(created at)'^^xsd:dateTime:
                                                                               fibo-fnd-acc-cur:hasCurrency <cb/>cb/currency/(valuation price currency code
 cb:updatedAt 'fixDate(updated_at)'^^xsd:dateTime.
                                                                             <cb/ipo/(uuid)/marketCapValue/USD> a fibo-fnd-acc-cur:MonetaryAmount;
<cb/ipo/(uuid)/share> a fibo-sec-sec-lst:ListedSecurity, fibo-sec-eq-eq:Share)
                                                                              rdfs:label "Monetary amount of Market Cap (valuation) in USD";
  rdfs:label "Share/listed security '(stock_symbol)'";
                                                                               fibo-fnd-acc-cur:hasAmount '(valuation price_usd)'^^xsd:decimal;
  fibo-sec-sec-lst:isListedVia <cb/ipo/(uuid)/listing>;
                                                                               fibo-fnd-acc-cur:hasCurrency <cb/currency/USD>.
  fibo-fnd-agr-ctr:hasPrincipalParty <cb/agent/(org uuid)/issuer>;
                                                                                                                                                           lcc-lr:isMemberOf <cb/currency>.
  fibo-fnd-rel-rel:isIssuedBy <cb/agent/(org uuid)/issuer>:
                                                                             <cb/ipo/(uuid)/pricePerShare/(share price currency code)> a fibo-sec-eq-eq+
  fibo-fbc-fi-fi:isDenominatedIn <cb/currency/(share price currency code)>
                                                                              rdfs:label "Price per share at the time of IPO (permalink) in local curred
  fibo-sec-sec-lst:hasOriginalPlaceOfListing <cb/exchange/(stock exchange :
                                                                               fibo-fnd-acc-cur:hasAmount '(share price)'^^xsd:decimal:
  fibo-sec-sec-lst:hasHomeExchange <cb/exchange/(stock exchange symbol)>.
                                                                               fibo-fnd-acc-cur:hasCurrency <cb/currency/(share price currency code)>;
                                                                                                                                                           rdfs:label 'CrunchBase currency code set';
                                                                               fibo-fnd-dt-fd:hasObservedDateTime '(went_public_on)'^^xsd:date;
<cb/ipo/(uuid)/listing> a fibo-sec-sec-lst:Listing;
                                                                               fibo-fnd-acc-cur:isPriceFor <cb/ipo/(uuid)/share>.
  rdfs:label "Listing of share/security '(stock_symbol)' on exchange '(stor)
  fibo-sec-sec-lst:lists <cb/ipo/(uuid)/share>;
                                                                             <cb/ipo/(uuid)/pricePerShare/USD> a fibo-sec-eq-eq:PricePerShare;
  Ton Tonda Tidone & Clad Don dala (for 1/ out d) (b) alsons
                                                                               adfa-label "Dadas are about at the time of TDO (secondials) in USO"
```

```
rdfs:label "Price per share at the time of IPO (permalink) in USD";
 fibo-fnd-acc-cur:hasAmount '(share_price_usd)'^^xsd:decimal;
 fibo-fnd-acc-cur:hasCurrency <cb/currency/USD>;
 fibo-fnd-dt-fd:hasObservedDateTime '(went_public_on)'^^xsd:date:
 fibo-fnd-acc-cur:isPriceFor <cb/ipo/(uuid)/share>.
## Currencies: USD plus 3 more for the 3 financials
<cb/currency/USD> a fibo-fnd-acc-cur:Currency;
<cb/currency/USD/code> a fibo-fnd-acc-cur:CurrencyIdentifier;
<cb/currency/(share price currency code)> a fibo-fnd-acc-cur:Currency;
 lcc-lr:hasName '(share price currency code)';
 rdfs:label 'Currency (share price currency code)'.
<cb/>cb/currency/(share price currency code)/code> a fibo-fnd-acc-cur:Currency>
 rdfs:label 'Code of currency (share price currency code)';
 lcc-lr:hasTag '(share price currency code)';
 lcc-ln:denotes <cb/currency/(share price currency code)>;
 lcc-lr:identifies <cb/currency/(share_price_currency_code)>;
<cb/currency/(valuation price currency code)> a fibo-fnd-acc-cur:Currency;
 lcc-lr:hasName '(valuation price currency code)';
 rdfs:label 'Currency (valuation price currency code)'.
ccb/currency/(valuation price currency code)/code> a fibo-fnd-acc-cur:Curre
 rdfs:label 'Code of currency (valuation price currency code)';
 lcc-lr:hasTag '(valuation_price_currency_code)';
 lcc-lr:denotes <cb/currency/(valuation_price_currency_code)>;
 lcc-lr:identifies <cb/currency/(valuation_price_currency_code)>;
<cb/currency/(money_raised_currency_code)> a fibo-fnd-acc-cur:Currency;
 lcc-lr:hasName '(money raised currency code)';
 rdfs:label 'Currency (money raised currency code)'.
ccb/currency/(money raised currency code)/code> a fibo-fnd-acc-cur:Currence
 rdfs:label 'Code of currency (money_raised_currency_code)';
 lcc-lr:hasTag '(money raised currency code)';
 lcc-lr:denotes <cb/currency/(money_raised_currency_code)>;
 lcc-lr:identifies <cb/currency/(money raised currency code)>;
<cb/currency> a lcc-lr:IdentificationScheme, lcc-lr:CodeSet;
 skos:definition 'Currency identifiers used in CrunchBase':
 skos:scopeNote 'Better reconcile to fibo-fnd-acc-4217:ISO4217-CodeSet'.
```

#### IPOs in FIBO: SPARQL (220 lines, 56 binds, 21 prefixes)

```
prefix cb: <https://ontotext.com/crunchbase/ontology/>
prefix fibo-fbc-fct-mkt: <a href="https://spec.edmcouncil.org/fibo/ontology/FBC/FunctionalEnt">https://spec.edmcouncil.org/fibo/ontology/FBC/FunctionalEnt</a>
                                                                                                                         lcc-lr:hasTag ?stock_symbol;
prefix fibo-fbc-fct-ra: <a href="https://spec.edmcouncil.org/fibo/ontology/FBC/FunctionalEnti">https://spec.edmcouncil.org/fibo/ontology/FBC/FunctionalEnti</a>
prefix fibo-fbc-fi-fi: <a href="https://spec.edmcouncil.org/fibo/ontology/FBC/financialInstru">https://spec.edmcouncil.org/fibo/ontology/FBC/financialInstru</a>
prefix fibo-fbc-pas-fpas: <a href="https://spec.edmcouncil.org/fibo/ontology/FBC/ProductsAndS">prefix fibo-fbc-pas-fpas: <a href="https://spec.edmcouncil.org/fibo/ontology/FBC/ProductsAndS">https://spec.edmcouncil.org/fibo/ontology/FBC/ProductsAndS</a>
prefix fibo-fnd-acc-cur: <a href="https://spec.edmcouncil.org/fibo/ontology/FND/Accounting/Cu">https://spec.edmcouncil.org/fibo/ontology/FND/Accounting/Cu</a>
prefix fibo-fnd-agr-ctr: <a href="https://spec.edmcouncil.org/fibo/ontology/FND/Agreements/Co">https://spec.edmcouncil.org/fibo/ontology/FND/Agreements/Co</a>
prefix fibo-fnd-arr-id: <a href="https://spec.edmcouncil.org/fibo/ontology/FND/Arrangements/l">https://spec.edmcouncil.org/fibo/ontology/FND/Arrangements/l</a>
prefix fibo-fnd-dt-fd: <a href="https://spec.edmcouncil.org/fibo/ontology/FND/DatesAndTimes/F">prefix fibo-fnd-dt-fd: <a href="https://spec.edmcouncil.org/fibo/ontology/FND/DatesAndTimes/F">https://spec.edmcouncil.org/fibo/ontology/FND/DatesAndTimes/F</a>)
prefix fibo-fnd-rel-rel: <a href="https://spec.edmcouncil.org/fibo/ontology/FND/Relations/Rel">prefix fibo-fnd-rel-rel: <a href="https://spec.edmcouncil.org/fibo/ontology/FND/Relations/Rel">https://spec.edmcouncil.org/fibo/ontology/FND/Relations/Rel</a>
prefix fibo-fnd-utl-alx: <a href="https://spec.edmcouncil.org/fibo/ontology/FND/Utilities/Ana">https://spec.edmcouncil.org/fibo/ontology/FND/Utilities/Ana
prefix fibo-ind-mkt-bas: <a href="https://spec.edmcouncil.org/fibo/ontology/IND/MarketIndices">https://spec.edmcouncil.org/fibo/ontology/IND/MarketIndices</a>
prefix fibo-sec-eq-eq: <a href="https://spec.edmcouncil.org/fibo/ontology/SEC/Equities/Equity">https://spec.edmcouncil.org/fibo/ontology/SEC/Equities/Equity</a>
prefix fibo-sec-sec-id: <a href="https://spec.edmcouncil.org/fibo/ontology/SEC/Securities/Sec">https://spec.edmcouncil.org/fibo/ontology/SEC/Securities/Sec
prefix fibo-sec-sec-iss: <a href="https://spec.edmcouncil.org/fibo/ontology/SEC/Securities/Se">https://spec.edmcouncil.org/fibo/ontology/SEC/Securities/Se</a>
prefix fibo-sec-sec-lst: <a href="https://spec.edmcouncil.org/fibo/ontology/SEC/Securities/Se">https://spec.edmcouncil.org/fibo/ontology/SEC/Securities/Se</a>
prefix lcc-lr: <https://www.omg.org/spec/LCC/Languages/LanguageRepresentation/>
prefix puml: <a href="mailto://plantuml.com/ontology#">
prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>
prefix skos: <a href="http://www.w3.org/2004/02/skos/core#">http://www.w3.org/2004/02/skos/core#></a>
prefix xsd: <http://www.w3.org/2001/XMLSchema#>
clear silent graph <cb/ipos>;
insert {graph <cb/ipos> {
  ## Issuer as role of a CB agent
  ?cb_agent_org_uuid_issuer_URL a fibo-fbc-fi-fi:Issuer, fibo-fbc-pas-fpas:Offeror;
    fibo-fnd-rel-rel:issues ?cb_ipo_uuid_share_URL;
    fibo-fnd-rel-rel:hasIdentity ?cb agent org uuid URL.
  ## Stock exchange
   ?cb_exchange_stock_exchange_symbol_URL a fibo-fbc-fct-mkt:Exchange, fibo-fbc-fct-ra>
    rdfs:label "Exchange ?stock exchange symbol";
    rdfs:comment "Unfortunately CB exchange codes are ambiguous, eg 'mse' could be Me >
     fibo-fbc-fct-mkt:hasExchangeAcronym ?stock exchange symbol;
    lcc-lr:isIdentifiedBy ?cb exchange stock exchange symbol code URL.
   ?cb exchange stock exchange symbol code URL a lcc-lr:Identifier:
    rdfs:label "Identifier of exchange ?stock exchange symbol";
    lcc-lr:hasTag ?stock_exchange_symbol;
    lcc-lr:identifies ?cb_exchange_stock_exchange_symbol_URL;
    lcc-lr:isMemberOf <cb/exchange/code>.
   <cb/exchange/code> a lcc-lr:CodeSet;
    rdfs:label 'CrunchBase exchange code scheme';
    rdfs:comment 'Unfortunately CB Exchange codes are ambiguous, unlike ISO 10383 MIC+
   ## Offering, Share, Listing, Ticker
   ?cb_ipo_uuid_offering_URL a fibo-sec-sec-iss:PublicOffering;
    rdfs:label "IPO (permalink)";
     fibo-fbc-pas-fpas:hasOfferingPrice ?cb_ipo_uuid_price_money_raised_currency_code_>
     fibo-fnd-rel-rel:appliesTo ?cb ipo uuid share URL;
     fibo-fnd-rel-rel:isIssuedBy 7cb agent org uuid issuer URL:
     fibo-sec-sec-iss:hasFirstTradeDate ?went public on xsd date:
     fibo-sec-sec-iss:isRegisteredWith ?cb exchange stock exchange symbol URL:
                                                                                                                        1cc-1r:hasName 'USD':
    cb:uuid ?uuid:
                                                                                                                        rdfs:label 'Currency USD'.
    cb:name ?name:
    cb:permalink ?permalink;
    cb:url ?cb_url_xsd_anyURI;
                                                                                                                        lcc-lr:hasTag 'USD';
    cb:rank ?rank_xsd_integer;
    cb:createdAt ?created_at_FIXDATE_xsd_dateTime;
    cb:updatedAt ?updated_at_FIXDATE_xsd_dateTime.
                                                                                                                        lcc-lr:isMemberOf <cb/currency>.
   ?cb_ipo_uuid_share_URL a fibo-sec-sec-lst:ListedSecurity, fibo-sec-eq-eq:Share;
    rdfs:label "Share/listed security ?stock symbol";
     fibo-sec-sec-lst:isListedVia ?cb_ipo_uuid_listing_URL;
     fibo-fnd-agr-ctr:hasPrincipalParty ?cb agent org uuid issuer URL;
     fibo-fnd-rel-rel:isIssuedBy ?cb agent org uuid issuer URL:
     fibo-fbc-fi-fi:isDenominatedIn ?cb currency share price currency code URL:
     fibo-sec-sec-lst:hasOriginalPlaceOfListing ?cb exchange stock exchange symbol URL>
    fibo-sec-sec-lst:hasHomeExchange ?cb_exchange_stock_exchange_symbol_URL.
   ?cb_ipo_uuid_listing_URL a fibo-sec-sec-lst:Listing;
    rdfs:label "Listing of share/security ?stock_symbol on exchange ?stock_exchange_s>
    fibo-sec-sec-1st:lists ?cb_ipo_uuid_share_URL;
     lcc-lr:isIdentifiedBy ?cb_ipo_uuid_ticker_URL;
     fibo-sec-sec-1st:isTradedOn ?cb_exchange_stock_exchange_symbol_URL;
     fibo-fnd-acc-cur:hasCurrency ?cb currency share price currency code URL.
```

```
rdfs:label "Ticker symbol/security identifier '(stock exchange symbol):(stock symbol)
 fibo-fbc-fct-ra:isRegisteredBy ?cb exchange stock exchange symbol URL:
 lcc-lr:identifies ?cb ipo uvid listing URL:
 fibo-fnd-arr-id:hasInitialAssignmentDate ?went public on xsd date.
?cb ipo uuid price money raised currency code URL a fibo-fnd-acc-cur:MonetaryPrice:>
 rdfs:label "Total price/money raised of IPO (permalink) in local currency";
 fibo-fnd-acc-cur:hasAmount ?money_raised_xsd_decimal;
 fibo-fnd-acc-cur:hasCurrency ?cb_currency_money_raised_currency_code_URL.
?cb_ipo_uuid_price_USD_URL a fibo-fnd-acc-cur:MonetaryPrice;
 rdfs:label "Total price/money raised of IPO (permalink) in USD";
 fibo-fnd-acc-cur:hasAmount ?money_raised_usd_xsd_decimal;
 fibo-fnd-acc-cur:hasCurrency <cb/currency/USD>.
?cb_ipo_uuid_marketCap_valuation_price_currency_code_URL a fibo-ind-mkt-bas:MarketC>
 rdfs:label "Market Cap (valuation) at the time of IPO (permalink) in local current
 fibo-ind-mkt-bas:hasMarketCapitalizationValue ?cb_ipo_uuid_marketCapValue_valuati } }}
 fibo-fnd-utl-alx:hasArgument ?cb_ipo_uuid_pricePerShare_share_price_currency_code where {
 fibo-fnd-dt-fd:hasObservedDateTime ?went public on xsd date:
 fibo-fnd-rel-rel:appliesTo ?cb_agent_org_uuid_issuer_URL.
?cb ipo uuid marketCap USD URL a fibo-ind-mkt-bas:MarketCapitalization;
 rdfs:label "Market Cap (valuation) at the time of IPO (permalink) in USD":
 fib ind-mkt-bas:hasMarketCapitalizationValue ?cb_ipo_uuid_marketCapValue_USD_URL
 fibo-fnd-utl-alx:hasArgument ?cb_ipo_uuid_pricePerShare_USD_URL;
 fibo-fnd-dt-fd:hasObservedDateTime ?went_public_on_xsd_date;
 fibo-fnd-rel-rel:appliesTo ?cb_agent_org_uuid_issuer_URL.
?cb ipo uuid marketCapValue valuation price currency code URL a fibo-fnd-acc-cur:Mo>
 rdfs:label "Monetary amount of Market Cap (valuation) in local currency";
 fibo-fnd-acc-cur:hasAmount ?valuation_price_xsd_decimal;
 fibo-fnd-acc-cur:hasCurrency ?cb_currency_valuation_price_currency_code_URL.
?cb_ipo_uuid_marketCapValue_USD_URL a fibo-fnd-acc-cur:MonetaryAmount;
 rdfs:label "Monetary amount of Market Cap (valuation) in USD";
 fibo-fnd-acc-cur:hasAmount ?valuation price usd xsd decimal;
 fibo-fnd-acc-cur:hasCurrency <cb/currency/USD>.
?cb ipo uuid pricePerShare share price currency code URL a fibo-sec-eq-eq:PricePerS*
 rdfs:label "Price per share at the time of IPO (permalink) in local currency";
 fibo-fnd-acc-cur:hasAmount ?share_price_xsd_decimal;
 fibo-fnd-acc-cur:hasCurrency ?cb_currency_share_price_currency_code_URL;
 fibo-fnd-dt-fd:hasObservedDateTime ?went_public_on_xsd_date;
 fibo-fnd-acc-cur:isPriceFor ?cb ipo uuid share URL.
?cb_ipo_uuid_pricePerShare_USD_URL a fibo-sec-eq-eq:PricePerShare;
 rdfs:label "Price per share at the time of IPO (permalink) in USD";
 fibo-fnd-acc-cur:hasAmount ?share_price_usd_xsd_decimal;
 fibo-fnd-acc-cur:hasCurrency <cb/currency/USD>;
 fibo-fnd-dt-fd:hasObservedDateTime ?went public on xsd date;
 fibo-fnd-acc-cur:isPriceFor ?cb_ipo_uuid_share_URL.
## Currencies: USD plus 3 more for the 3 financials
<cb/currency/USD> a fibo-fnd-acc-cur:Currency:
<cb/currency/USD/code> a fibo-fnd-acc-cur:CurrencyIdentifier;
 rdfs:label 'Code of currency USD';
 lcc-lr:denotes <cb/currency/USD>;
 lcc-lr:identifies <cb/currency/USD>;
?cb_currency_share_price_currency_code_URL a fibo-fnd-acc-cur:Currency;
 1cc-lr:hasName ?share price currency code;
 rdfs:label 'Currency (share_price_currency_code)'.
?cb currency share price currency code code URL a fibo-fnd-acc-cur:CurrencyIdentifi>
 rdfs:label 'Code of currency (share price currency code)';
 lcc-lr:hasTag ?share price currency code;
 lcc-lr:denotes ?cb currency share price currency code URL:
 lcc-lr:identifies ?cb_currency_share_price_currency_code_URL;
 lcc-lr:isMemberOf <cb/currency>.
?cb_currency_valuation_price_currency_code_URL a fibo-fnd-acc-cur:Currency;
 lcc-lr:hasName ?valuation_price_currency_code;
 rdfs:label 'Currency (valuation_price_currency_code)'.
?cb_currency_valuation_price_currency_code_code_URL a fibo-fnd-acc-cur:CurrencyIden>
 rdfs:label 'Code of currency (valuation_price_currency_code)';
```

```
1cc-lr:denotes ?cb_currency_valuation_price_currency_code_URL;
 lcc-lr:identifies ?cb currency valuation price currency code URL;
 lcc-lr:isMemberOf <cb/currency>.
?cb currency money raised currency code URL a fibo-fnd-acc-cur:Currency;
 1cc-lr:hasName ?money raised currency code:
 rdfs:label 'Currency (money raised currency code)'.
?cb currency money raised currency code code URL a fibo-fnd-acc-cur:CurrencyIdentif>
 rdfs:label 'Code of currency (money_raised_currency_code)';
 lcc-lr:hasTag ?money_raised_currency_code;
 lcc-lr:denotes ?cb_currency_money_raised_currency_code_URL;
 lcc-lr:identifies ?cb_currency_money_raised_currency_code_URL;
 lcc-lr:isMemberOf <cb/currency>.
<cb/currency> a lcc-lr:IdentificationScheme, lcc-lr:CodeSet;
 rdfs:label 'CrunchBase currency code set';
 skos:definition 'Currency identifiers used in CrunchBase';
 skos:scopeNote 'Better reconcile to fibo-fnd-acc-4217:ISO4217-CodeSet'.
service <rdf-mapper:ontorefine:PROJECT ID> {
 bind(?c ore uuid as ?ore uuid)
 bind(?c uuid as ?uuid)
 bind(?c_stock_exchange_symbol as ?stock_exchange_symbol)
 bind(?c_permalink as ?permalink)
 bind(?c_money_raised_currency_code as ?money_raised_currency_code)
 bind(?c_went_public_on as ?went_public_on)
 bind(?c_name as ?name)
 bind(?c cb url as ?cb url)
 bind(?c_rank as ?rank)
 bind(?c_created_at as ?created_at)
 bind(?c updated at as ?updated at)
 bind(?c stock symbol as ?stock symbol)
 bind(?c share price currency code as ?share price currency code)
 bind(?c money raised as ?money raised)
 bind(?c money raised usd as ?money raised usd)
 bind(?c valuation price currency code as ?valuation price currency code)
 bind(?c_valuation as ?valuation)
 bind(?c_valuation_price as ?valuation_price)
 bind(?c_valuation_price_usd as ?valuation_price_usd)
 bind(?c_share_price as ?share_price)
 bind(?c_share_price_usd as ?share_price_usd)
 bind(iri(concat("cb/agent/",?org_uuid,"/issuer")) as ?cb_agent_org_uuid_issuer_UR>
 bind(iri(concat("cb/ipo/",?uuid,"/share")) as ?cb_ipo_uuid_share_URL)
 bind(iri(concat("cb/agent/",?org_uuid)) as ?cb_agent_org_uuid_URL)
 bind(iri(concat("cb/exchange/",?stock_exchange_symbol)) as ?cb_exchange_stock_exc>
 bind(iri(concat("cb/exchange/",?stock_exchange_symbol,"/code")) as ?cb_exchange_s>
 bind(iri(concat("cb/ipo/", ?uuid, "/offering")) as ?cb_ipo_uuid_offering_URL)
 bind(iri(concat("cb/ipo/",?uuid,"/price/",?money_raised_currency_code)) as ?cb_ip>
 bind(iri(concat("cb/ipo/",?uuid,"/price/USD")) as ?cb_ipo_uuid_price_USD_URL)
 bind(strdt(?went_public_on,xsd:date) as ?went_public_on_xsd_date)
 bind(strdt(?cb_url,xsd:anyURI) as ?cb_url_xsd_anyURI)
 bind(strdt(?rank,xsd:integer) as ?rank_xsd_integer)
 bind(REPLACE(?created_at,' ','T') as ?created_at_FIXDATE)
bind(strdt(?created_at_FIXDATE,xsd:dateTime) as ?created_at_FIXDATE_xsd_dateTime);
 bind(REPLACE(?updated_at,' ','T') as ?updated_at_FIXDATE)
bind(strdt(?updated_at_FIXDATE,xsd:dateTime) as ?updated_at_FIXDATE_xsd_dateTime)
}
 bind(iri(concat("cb/ipo/",?uuid,"/listing")) as ?cb_ipo_uuid_listing_URL)
 bind(iri(concat("cb/currency/",?share_price_currency_code)) as ?cb_currency_share>
 bind(iri(concat("cb/ipo/",?uuid,"/ticker")) as ?cb_ipo_uuid_ticker_URL)
 bind(strdt(?money_raised,xsd:decimal) as ?money_raised_xsd_decimal)
 bind(iri(concat("cb/currency/",?money_raised_currency_code)) as ?cb_currency_mone>
 bind(strdt(?money raised usd.xsd:decimal) as ?money raised usd xsd decimal)
 bind(iri(concat("cb/ipo/",?uuid,"/marketCap/",?valuation_price_currency_code)) as>
 bind(iri(concat("cb/ipo/",?uuid,"/marketCapValue/",?valuation price currency code
 bind(iri(concat("cb/ipo/",?uuid,"/pricePerShare/",?share_price_currency_code)) as>
 bind(iri(concat("cb/ipo/",?uuid,"/marketCap/USD")) as ?cb_ipo_uuid_marketCap_USD_>
 bind(iri(concat("cb/ipo/",?uuid,"/marketCapValue/USD")) as ?cb_ipo_uuid_marketCap>
 bind(iri(concat("cb/ipo/", ?uuid, "/pricePerShare/USD")) as ?cb_ipo_uuid_pricePerShare/USD")) as ?cb_ipo_uuid_pricePerShare/USD"))
 bind(strdt(?valuation_price,xsd:decimal) as ?valuation_price_xsd_decimal)
 bind(iri(concat("cb/currency/",?valuation_price_currency_code)) as ?cb_currency_v)
 bind(strdt(?valuation price usd,xsd:decimal) as ?valuation price usd xsd decimal)?
```

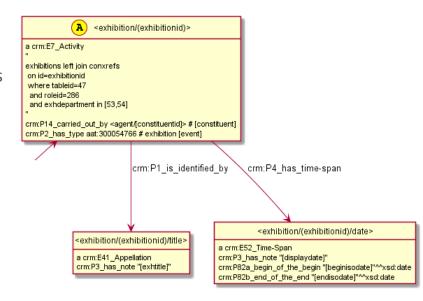
# **Generating R2RML Transforms**



# R2RML Transforms From Turtle Examples

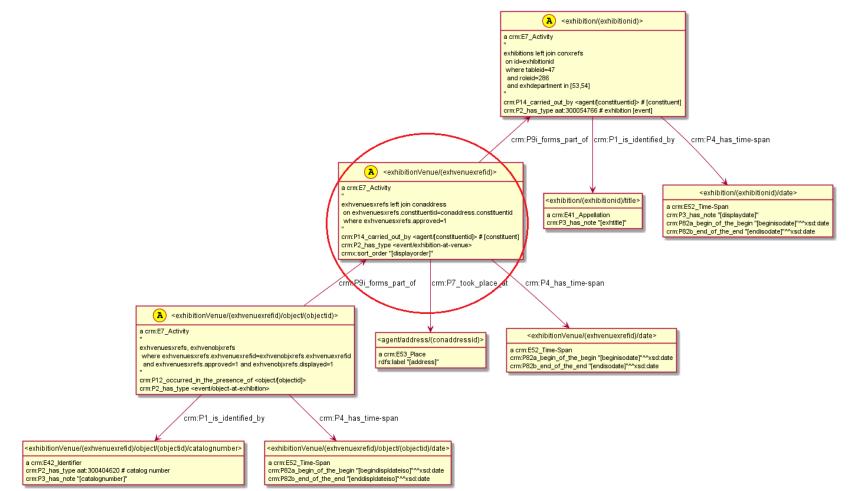
- R2RML: W3C language to map RDBMS to RDF
- rdf2rml: generates R2RML transforms from Turtle examples
  - Table/join in node comment; carried to "children" nodes
  - Field in attribute/URL
  - Both the model and R2RML are RDF, so implemented as some SPARQL transformation
  - TODO: implement functions
  - Example: Exhibitions from J.P.Getty Museum
- Then can use any R2RML implementation:
  - ONTOP for RDBMS Virtualization (OBDA)
  - GraphDB Virtual Repositories for Virtualization or Materialization
  - Various implementations, see <u>awesome-semantic-web:</u> <u>r2rml</u>:
    Antidot d2btriples, Morph-KGC, RMLmapper.

Antidot d2btriples, Morph-KGC, RMLmapper, RocketRML, carml...

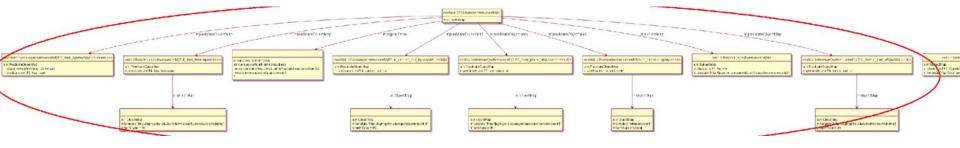




### Getty Exhibitions: Turtle model



## Getty Exhibitions: Generated R2RML



- R2RML is quite verbose:
  - subject map,
  - predicateObject map,
  - Templates and details about each URL and literal, etc.
- The one circled node is expanded to 15 nodes



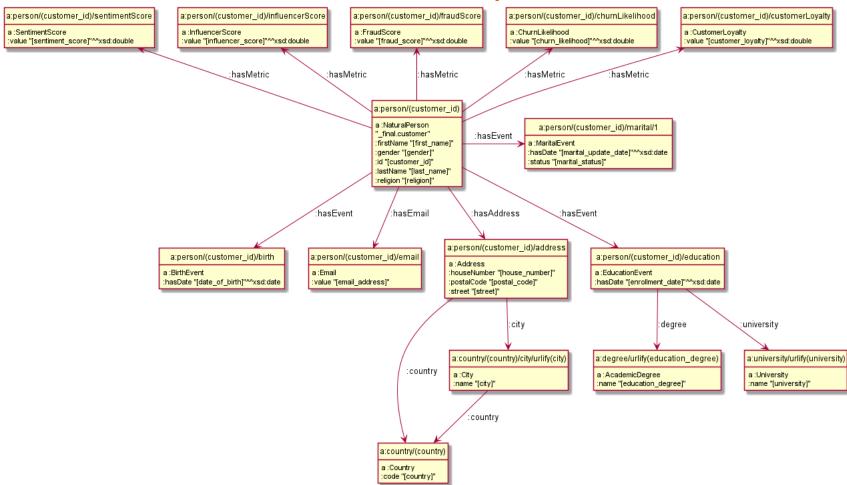
#### Getty Exhibitions: Model (60 lines) vs R2RML (315 lines)

```
## LEVEL 2: EXHIBITION AT VENUE
# An exhibition may visit a venue more than once: MUSLOD-1
<exhibitionVenue/(exhvenuexrefid)>
 puml:label """
exhvenuesxrefs left join conaddress
 on exhvenuesxrefs.constituentid=conaddress.constituentid
 where exhvenuesxrefs.approved=1
  a crm: E7_Activity;
  crm:P2_has_type <event/exhibition-at-venue>;
  crm:P9i forms part of <exhibition/(exhibitionid)>;
  crm:P14 carried_out_by <agent/(constituentid)>;
  crm:P7 took place at <agent/address/(conaddressid)>;
  crm:P4 has time-span <exhibitionVenue/(exhvenuexrefid)/date>;
  crmx:sort order "(displayorder)".
<exhibitionVenue/(exhvenuexrefid)/date> a crm:E52 Time-Span;
  crm:P82a begin of the begin "(beginisodate)"^^xsd:date;
  crm:P82b end of the end "(endisodate)"^^xsd:date.
<agent/(constituentid)> a crm:E39 Actor;
  rdfs:label "(constituent)".
<agent/address/(conaddressid)> a crm:E53 Place;
  rdfs:label "(address)".
## LEVEL 3: OBJECT AT EXHIBITION AT VENUE
<exhibitionVenue/(exhvenuexrefid)/object/(objectid)>
 puml:label """
exhvenuesxrefs, exhvenobixrefs
 where exhvenuesxrefs.exhvenuexrefid=exhvenobjxrefs.exhvenuexrefid
  and exhvenuesxrefs.approved=1 and exhvenobjxrefs.displayed=1
  a crm: E7 Activity;
  crm:P2 has type <event/object-at-exhibition>;
  crm: P9i forms part of <exhibitionVenue/(exhvenuexrefid)>;
  crm:P12_occurred_in_the_presence_of <object/(objectid)>;
  crm:P4_has_time-span <exhibition(enue/(exhvenuexrefid)/object/(objectid),
  crm:P1 is identified by <exhibitionVenue/(exhvenuexrefid)/object/(object:
<exhibitionVenue/(exhvenuexrefid)/object/(objectid)/date> a crm:E52 Time-Sp
  crm: P82a_begin_of_the_begin "(begindispldateiso)"^^xsd:date;
  crm:P82b end of the end "(enddispldateiso)"^^xsd:date.
<exhibitionVenue/(exhvenuexrefid)/object/(objectid)/catalognumber> a crm:E/
  crm:P2 has type aat:300404620; # catalog number: ITSLOD-466
  crm:P3 has note "(catalognumber)".
*******
aat:300054766 a puml:Inline; skos:prefLabel "exhibition (event)".
aat:300404620 a puml:Inline; skos:prefLabel "catalog number".
crm: P9i forms part of puml:arrow puml:up.
<agent/(constituentid)> a puml:Inline.
<event/exhibition-at-venue> a puml:Inline.
<event/object-at-exhibition> a puml:Inline.
```

```
kagent/address/(conaddressid)!label!(address)>
                    nn:PredicateObjectMap;
       rr:objectMap [ a
                                  rr:ObjectMap ;
                       rr:template "{address}" :
                      rr:termType rr:Literal
       rr:predicate rdfs:label .
<agent/address/(conaddressid)!map>
                             rr:TriplesMap ;
                             [ rr:sqlQuery "select * from
       rr:logicalTable
\nexhvenuesxrefs left join conaddress \n on
exhvenuesxrefs.constituentid=conaddress.constituentid \n where
exhvenuesxrefs.approved=1\n"];
       rr:predicateObjectMap
<agent/address/(conaddressid)!label!(address)>;
       rr:subjectMap
                             <agent/address/(conaddressid)!subi> .
<agent/address/(conaddressid)!subj>
                   rr:SubjectMap :
                   crm: E53 Place ;
       rr:template "agent/address/{conaddressid}" .
<exhibition/(exhibitionid)!P14 carried out by!(constituentid)>
                    rr:PredicateObjectMap :
       rr:objectMap [ a
                                  rr:ObjectMap;
                      rr:template "agent/{constituentid}";
                      rr:termType rr:IRI
                    ];
       rr:predicate crm:P14 carried out by .
<exhibition/(exhibitionid)!P1_is_identified_by!title>
                    rr:PredicateObjectMap :
       rr:objectMap [ a
                                  rr:ObjectMap :
                      rr:template "exhibition/{exhibitionid}/title";
                      rr:termType rr:IRI
       rr:predicate crm:P1 is identified by .
<exhibition/(exhibitionid)!P2 has type!300054766>
                    rr:PredicateObjectMap :
       rr:object
                    aat:300054766 :
       rr:predicate crm:P2 has type .
<exhibition/(exhibitionid)!P4 has time-span!date>
                    rr:PredicateObjectMap :
       rr:objectMap [ a
                                  rr:ObjectMap ;
                      rr:template "exhibition/{exhibitionid}/date" :
                      rr:termType rr:IRI
                    1;
       rr:predicate crm:P4_has_time-span .
<exhibition/(exhibitionid)!map>
                             rr:TriplesMap :
       rr:logicalTable
                             [ rr:sqlQuery "select * from
\nexhibitions left join conxrefs \n on id=exhibitionid\n where
```

```
rr:logicalTable
                             [ rr:sqlQuery "select * from
\nexhibitions left join conxrefs \n on id=exhibitionid\n where
Hableid=47\n and roleid=286 \n and exhdepartment in (53,54)\n" ];
       rr:predicateObjectMap
<exhibition/(exhibitionid)!P1 is identified by!title> .
<exhibition/(exhibitionid)!P4 has time-span!date> ,
<exhibition/(exhibitionid)!P14 carried out by!(constituentid)> .
<exhibition/(exhibitionid)!P2 has type!300054766> ;
       rr:subjectMap
                             <exhibition/(exhibitionid)!subj> .
<exhibition/(exhibitionid)!subj>
                   rr:SubjectMap :
                   crm: E7 Activity ;
       rr:class
       rr:template "exhibition/{exhibitionid}" .
<exhibition/(exhibitionid)/date!P3 has note!(displaydate)>
                     rr:PredicateObjectMap :
       rr:objectMap [ a
                                   rr:ObjectMap ;
                       rr:template "{displaydate}";
                      rr:termType rr:Literal
       rr:predicate crm:P3_has_note .
<exhibition/(exhibitionid)/date!P82a begin of the begin!(beginisodate)>
                     rr:PredicateObjectMap :
       rr:objectMap [ a
                                   rr:ObjectMap ;
                       rr:datatype xsd:date;
                      rr:template "{beginisodate}";
                       rr:termType rr:Literal
       rr:predicate crm:P82a begin of the begin .
<exhibition/(exhibitionid)/date!P82b_end_of_the_end!(endisodate)>
                     rr:PredicateObjectMap;
       rr:objectMap [ a
                                   rr:ObjectMap :
                       rr:datatype xsd:date;
                      rr:template "{endisodate}";
                      rr:termType rr:Literal
       rr:predicate crm:P82b end of the end .
<exhibition/(exhibitionid)/date!map>
                             rr:TriplesMap ;
       rr:logicalTable
                             [ rr:sqlQuery "select * from
\nexhibitions left join conxrefs \n on id=exhibitionid\n where
tableid=47\n and roleid=286 \n and exhdepartment in (53,54)\n" ];
       rr:predicateObjectMap
<exhibition/(exhibitionid)/date!P82b end of the end!(endisodate)> .
<exhibition/(exhibitionid)/date!P3 has note!(displaydate)>,
<exhibition/(exhibitionid)/date!P82a_begin_of_the_begin!(beginisodate)>;
       rr:subjectMap
                             <exhibition/(exhibitionid)/date!subj> .
<exhibition/(exhibitionid)/date!subj>
                   rr:SubjectMap;
       rr:class crm:E52_Time-Span;
```

## Insurance Customer Example: Model



# Insurance Example: Model (52 lines) vs R2RML (502 lines)

@base chttps://ekg.allianz.com/resource/>.

MARKARANANANANANANANANA customer

Section   Contemp.   Description   Descrip	HANNANANANANAN CUSTOMER	#prefix   chttps://ekg.ailianz.com/resource/5.	<pre><person (customer_id)!hasevent!birth=""></person></pre>	rr:predicate :city .	rr:logicalTable <person (customer_id)="" customer<="" th=""></person>
		#prefix puml: <a href="http://plantuml.com/ontology#">http://plantuml.com/ontology#&gt;.</a>	a rr:PredicateObjectNap :		rr:predicateObjectMap <pre></pre>
			rr:objectMap [ a rr:ObjectMap;	<pre>cperson/(customer_id)/address!country!(country)&gt;</pre>	rr:subjectHap cperson/(customer_id)/customer
Institute (last part)   Inst		Woreflx rdf: <a href="http://www.w3.org/1999/82/22-rdf-syntax-ns8">http://www.w3.org/1999/82/22-rdf-syntax-ns8</a> .	rr:tereType rr:IRI	problectNap I a problectNap :	(person/(customer_id)/customerLoyalty(sub1)
		@prefix xsd: chttp://www.w3.org/2001/XMLSchemall> .	1:	rr:template "country/{country}";	a rr:SubjectMap;
			rr:predicate :hasEvent .	rr:termType rr:IRI	
General Content   Display   Display   Content   Display   Display   Content   Display   Displa	:lastName "(last_name)";	a re:PredicateObjectMap:	cperson/(customer_ld)thasEventteducation>	rr:predicate :country .	Pritemptate person/(customer_to)/customertoyalty 9
Intelligence operator (Contemps_16) Anterests   1	:gender "(gender)";	erroblectNam I a reroblectNam r	a rr:PredicateObjectMap :		<pre><person (customer_id)="" customerloyalty!value!(customer_loyalty)<="" pre=""></person></pre>
Intelligation   Control (Control   1)   Possible   Control   Con	:religion "(religion)";	rr:template "country/{country}";	rr:objectMap [ a rr:ObjectMap;	<pre><pre><pre><pre><pre><pre><pre><pre></pre></pre></pre></pre></pre></pre></pre></pre>	a rr:PredicateObjectMap;
Passent (contemp (a) print to )  Proposition (co	:hasAddress <person (customer="" address="" id)="">:</person>		rr:template person/(customer_in)/education .	rrioblectNap ( a rriOblectNap :	rr:objectnap [ a rr:objectnap ; rr:datatype wsd:double ;
		rr:predicate :country .	1:	rr:template "{house number}";	rr:template "(customer_loyalty)";
Passage   Control (Control   A) Proceedings   A) Proceedings   A Proceedings   A) Proceedings   A) Proceedings   A Proceedings   A) Proceedings   A Proceedings   A) Proceedings   A Proceedings   A) Proceedings   A Proceedings   A)		constantinum territorium lifetici territorium lifetici del la constantinum	rr:predicate :hasEvent .		
Second (contemp.   3) / section   1   second   1   seco		rr:tableName "final.customer".	<pre><person (customer="" id)!hasmetric!churnlikelihood=""></person></pre>		
Shalleric coreon (catedors, 10) / respectively control (catedors, 10			a rr:PredicateObjectMap :	According to the Control of the Cont	The state of the s
Institution control (cattors (3) / provided and provided		<pre><country (country)="" city="" urlify(city)!map=""></country></pre>	rr:objectMap [ z rr:ObjectMap ;	<pre><person (customer_id)="" address(itable)<="" pre=""></person></pre>	<pre><person (customer_id)="" education!degree!urlify(education_degre)<="" pre=""></person></pre>
Institute (gerosn) (catacher_16) partial [1] Institute (gerosn) (c		rr:logicalTable <country (country)="" city="" td="" urlify(city))<=""><td>rr:termType rr:IRI</td><td>Tritablename _Timal.customer .</td><td>rr:objectMap [ a rr:ObjectMap :</td></country>	rr:termType rr:IRI	Tritablename _Timal.customer .	rr:objectMap [ a rr:ObjectMap :
# Topology of the control (context—id)/martial)  # Topology of the control (context—id)/martial)  ## Topology of the control (context—id)/martial)		rr:predicateObjectNap <country (country)="" city="" td="" urlify(city)(<=""><td>1:</td><td><pre><person (customer_id)="" address!map=""></person></pre></td><td>rr:template "degree/urlify{education</td></country>	1:	<pre><person (customer_id)="" address!map=""></person></pre>	rr:template "degree/urlify{education
as condenses as indirectly production of the pro	:hashetric (person/(customer_id)/influencerscore);		rr:predicate :hasMetric .		rr:termType rr:IRI
instruction (person) (catedra-(1)) effects a bidders)  Person (1) effects a b		Prisubject/up (country)/city/urlity(city)/	operson/(customer_id)(hasMetric(customertovalty)	rr:regicaliable cperson/(customer_lo)/address:ltab.	pripredicate idegree .
Geron/Coatener_10/neroes a lideres; "rising "r		<pre><country (country)="" clty="" urlify(clty)!name!(clty)=""></country></pre>	a rr:PredicateObjectNap;	rr:subjectMap cperson/(customer_id)/address!subj	
represent (customer_(a)) and responses a statement of the control (control	:hasEvent <person (customer_id)="" education="">.</person>	a rr:PredicateObjectMap;	rr:objectMap [ a rr:ObjectMap ;		<pre><person (customer_id)="" education!hasdate!(enrollment_date)=""></person></pre>
Triestic control (control)   Triestic control (control (c		rr:objectMap [ a rr:ObjectMap ;	rr:template "person/(customer_ld)/customerLoy	* (person/(customer_id)/address!postalcode!(postal_code))	a rr:PredicateObjectNap;
Size of (Street);		rr:termType rr:Literal	11	rr:objectMap [ a rr:ObjectMap ;	rridatatype Ksd:date:
control(control(control)(control)(control(control(control(control)))   control (control(control(control(control)))   control (control(contro		accommon 14 constant and a constant	rr:predicate :hasMetric .	rr:template "{postal code}";	rr:template "(enrollment date)" ;
country/(country)   country   coun		rr:predicate :name .	concessor//constoners 1d\lbacketels/ldcandScores		rr:termType rr:titeral
including   country (country) city/unitry(city)   country (country) country (country)   country (country	:postalCode "(postal_code)";	<pre><country (country)="" city="" subj="" urlify(city)=""></country></pre>	a rr:PredicateObjectMap :	rr:predicate :postalCode .	re:predicate :hasDate .
country/(country)-icty/urify(city) = (city; country/country/country/country/city/urify(city) = (city; country/country/city/urify(city) = (city; country/country/city/urify(city)) = (city; country/city/urify(city)) = (city; country/country/city/urify(city)) = (city; country/country/city/urify(city)) = (city; country/country/city/urify(city)) = (city; country/country/city/urify(city)) = (city) = (ci	:city <country (country)="" city="" urlify(city)="">:</country>	a rr:SubjectMap;	rr:objectNap [ a rr:ObjectNap ;		
### Country/(country) a (City) recountry (country)  ### Country/(country) a (City) recountry (country)  ### Country/(country) a (Country) **  ### Country (Country) **  ### Countr			rr:template "person/{customer_id}/fraudScore"	(person/(customer_id)/address!street!(street)>	<pre>(person/(customer_id)/education!!table)</pre>
Country/(country) a !Country (country) code "(country) code" (country) code "(country) code" (country) code (country) country		rr.temptate country/(country//tity/oritry(city) .	1:	rr:objectMap [ a rr:ObjectMap ]	Pricationse Timalifusconer ;
country/(country) = iCountry): code "(country)":	<pre><country (country)="" city="" city)="" uclify=""> a :City: :country <count></count></country></pre>	<pre><degree urlify(education_degree)!!table=""></degree></pre>	rr:predicate :hasMetric :	rr:template "{street}";	<pre><person (customer_id)="" education!map=""></person></pre>
Country  c	Accountry/(country//city/urilly(city/) a reley/ resultry (country)	rr:tableName "_final.customer".	The second secon	rr:termType rr:titeral	a rr:TriplesMap;
gerson/(customer_id)/pirth a::Birthbent; ihasDate "(date_of)  operson/(customer_id)/eachin_gerson/idea  operson/(c	(country/(country)) a (Country) scade "(country)"	(degree/urlify(education degree)(map)	cperson/(customer_io):masmetric:inviuencerscores a pr:PredicateObjectNap :	rr:predicate :street .	
Gerson/Customer_id/self> : identifyent; inside to repredict independent of the property of control of the property of the	(country) a reductry, reductry,	a rr:TriplesMap :	rr:objectMap   a rr:ObjectMap ;		rr:subjectMap cperson/(customer_id)/educatio
<pre>cprson/(customer_id)/postile = (tending-ig/postile) = (tending-</pre>	The second secon	rr:logicalTable <degree td="" urlify(education_degree)!!tal<=""><td>rr:template "person/(customer_ld)/influencerS</td><td></td><td>Control of the Control of the Contro</td></degree>	rr:template "person/(customer_ld)/influencerS		Control of the Contro
person/(customer_id)/customer/costaty a :final[; value "(email_address) or "respitate (emailine.digness) or "respitate (emailine.dig	<pre><person (customer_id)="" birth=""> a :BirthEvent; :nasuate "(date_of;)</person></pre>	rr:predicateObjectMap cdegree/urlify(education_degree)!name			<pre>cperson/(customer_id)/education(subj) a</pre>
Gerson/(customer_id)/customer_id)/sustainer_idopality is a person/customer_id)/sentimentscore a isentimentscore; is a person/customer_id)/sentimentscore a influencer_score a in					rr:class :EducationEvent;
<pre>gerson/(customer_id)/customersCore) as if routScore) (represent/customer_id)/settlementScore) as if routScore) (represent/customer_id)/fraudScore) (represent/customer_id)/fraudScore) (represent/customer_id)/fraudScore) (represent/customer_id)/fraudScore) (represent/customer_id)/fraudScore) (represent/customer_id)/fraudScore) (represent/customer_id)/influencerScore) (represent/customer_id)/influence</pre>	<pre><person (customer_id)="" email=""> a :Email; :value "(email_address))</person></pre>		and the second of the second o		rn:template "person/(customer_id)/education" .
**Sperson/(cutsomer_id)/familicalitodo a controlled from the control of the contr		a rr:FredicateObjectMap;	cperson/(customer_id) hasMetric sentimentScore>	concent/customer_fd\/higththasNate//data of higth\/a	concern // customer. Let) / aducat Les fundament l'extuel L'éu/ un lunes L'ex-
<pre>Gperson/(customer_id)/sentimentScore) a iSentimentScore) if returning returning person/(customer_id)/sentimentScore) is a frequency in a</pre>		rr:template "(education degree)";	rr:objectMap [ a rr:ObjectMap :	a rr:PredicateObjectMap :	a rr:PredicateOblectMap :
person/(customer_id)/churnikelihoods a :Churnikelihoods interpolations interpolations in the person (customer_id)/influenceScore) a :Churnikelihoods interpolation (customer_id)/influenceScore) a :Churnikelihoods interpo		rr:tersType rr:Literal	rr:template "person/{customer_id}/sentimentSc	rr:objectMap [ a rr:ObjectMap ;	rr:objectMap [ a rr:ObjectMap ;
<pre>deperson/(customer_id)/inclinationals a: Churnitikelimods a:</pre>	<pre><person (customer_id)="" fraudscore=""> a :FraudScore; :va )</person></pre>	managed to the large		rr:datatype wsd:date;	rr:template "university/urlify(unive)
<pre>depreson/(customer_id)/influencerScore) :ua</pre>	<pre><person (customer_id)="" churnlikelihood=""> a :ChurnLikelihood; :va)</person></pre>	er-pressure insee .			1:
cperson/(customer_id)/marital/1) a :NaritalEvent;	<pre><person (customer_id)="" influencerscore=""> a :InfluencerScore; :va&gt;</person></pre>	<pre><degree urlify(education_degree)!subj=""></degree></pre>		1.1	rr:predicate :university .
<pre>cyerson/(customer_id)/marital]/&gt; a :!MaritalEvent; istatus "(marital_supdate_date)" cyerson/(customer_id)/education a :EducationEvent; isablet "(enrollent_date)" none" ("involventy); isdepte (enrollent_date)" none "(involventy); isdepte (enrollent_date)" none "(involventy); isdepte (enrollent_date)" none "(involventy); isdepte (edepte/unify(education_degree)).  cuniversity/unify(university) a :University); isdepte (edepte/unify(education_degree)).  cuniversity/unify(university).  isdepte (edepte/unify(education_degree)).</pre>		a rr:SubjectMap :	<pre><person (customer_id)!id!(customer_id)=""></person></pre>	rr:predicate :hasDate .	second Contract Interestitities
istatus "(marital_status)"; inhasDate "(marital_update_date)"	<pre><pre><pre><pre><pre><pre><pre><pre></pre></pre></pre></pre></pre></pre></pre></pre>	rr:template "degree/urlifv(education degree)".	rr:objectMap 1 a rr:ObjectMap :	operson/(customer id)/birth!!table>	rr:tableName " final customer".
<pre>cperson/(customer_id)/education a :EducationEvent;</pre>			rr:template "{customer_id}";	rr:tableName "_final.customer".	
<pre>cycron/(customer_id)/education a :EducationEvent; insolned "enonlinent date" \</pre>	istatus (mai rear_status) , imasaate (mai rear_apaate_aate) ,	<pre><person (customer_id)!firstname!(first_name)=""></person></pre>		Commission and Commission of the large state of the	<pre>cperson/(customer_id)/email!map&gt;</pre>
riseplate "(enrollment_date)" "mysdcidate; printerplate "(irst, name)"; printerplate "(irst, name); printerplate "	(manager//gustaman id)(adusation) a (EdusationEurati	recoblectMan [ a recoblectMan ]		(person/(customer_id)/birth(map)	enviouscalTable coerson/(sustanen id)/envillin
inviversity university/university); idegree (degree/unify(education_degree)).  cuniversity (university) in a constraint of the constraint		rr:template "{first_name}";		rr:logicalTable <pre></pre>	rr:predicateObjectMap <pre>cperson/(customer_Id)/email!vc)</pre>
idegree <pre>idegree /degree/unlify(education_degree)&gt;.</pre> <pre>     respectable instance.</pre>		rr:termType rr:Literal	<pre><person (customer_id)!lastname!(last_name)=""></person></pre>	rr:predicateObjectMap <pre>cperson/(customer_id)/birth!hasDate</pre>	rr:subjectMap <pre></pre>
retemplate "(last_name"); retemplate param/(customer_la)/harditallellood) retemplate "(last_name"); retemplate param/(customer_la)/harditallellood) retemplate "(last_name"); retemplate param/(customer_la)/harditallellood) retemplate "(last_name"); retemplate param/(customer_la)/harditallellood) retemplate param/(customer_la)/harditallellood) retemplate "(last_name"); retemplate param/(customer_la)/harditallellood) retemplate param/(customer_la)/harditallellood) retemplate "(last_name"); retemplate param/(customer_la)/harditallellood) retemplate param/(customer_la)/hardita		respectivate relevanment	resphiestMan [ a resolvestMan :	rr:subjectMap <pre>cperson/(customer_1d)/birth(sub)&gt; *</pre>	coerson/(customer_id)/email1(subi)
<pre>cuniversity/unify(university) a :University; :name "(u</pre>	:degree <degree urlity(education_degree)="">.</degree>		rr:template "(last name)";	<pre>cperson/(customer_id)/birth!subj&gt;</pre>	a rr:SubjectMap :
<pre>cdegree/unlify(education_degree) a :AcademicDegree; :name "(e)</pre>	and the second s	<pre>cperson/(customer_id)!gender!(gender)&gt;</pre>		a rr:SubSectMap :	rr:class :Email ;
<pre>componed continue (continue (co</pre>		resoblectMap [ a resoblectMap :	pripredicate :lastName .		rr:cemptate "person/(customer_ld)/email".
##### plantuml instructions    rectangle recta	<pre><degree urlify(education_degree)=""> a :AcademicDegree; :name "(e)</degree></pre>	rr:template "{gender}";			<pre><person (customer_id)="" email!value!(email_address)=""></person></pre>
respond(customer_id) thankfarilineall) respond(customer_id) hankfarilineall) respond(customer_id) hankfarilineall respond(customer_i			cperson/(customer_id)!ltable>	<pre><pre><pre><pre><pre><pre><pre><pre></pre></pre></pre></pre></pre></pre></pre></pre>	a pr:PredicateObjectMap ;
chashetric puml:arrow puml:up.  cprson/(customer_id) hashAddress!address)	##### plantuml instructions	reconditive conden	rr:tableName "_final.customer".	rr:tableName "_final.customer".	rn:objectNap [ a rn:objectNap ;
composition of the proof of the	Control National Control Contr	A CONTRACTOR OF THE CONTRACTOR	<pre>cperson/(customer id)!map&gt;</pre>	<pre>cperson/(customer id)/churnLikelihoodimap&gt;</pre>	rr:termType rr:Literal
resplecting [a probjecting are problected and problecting are problected and problems. The problecting are problected and problems are problected and problems are problected and problems. The problems are problems are problected and problems are problems are problems. The problems are problems are problems are problems are problems are problems. The problems are problems are problems are problems are problems are problems. The problems are problems are problems are problems are problems are problems. The problems are problems are problems are problems are problems are problems. The problems are problems are problems are problems are problems are problems. The problems are problems are problems are problems are problems are problems. The problems are problems are problems are problems are problems are problems. The problems are problems are problems are problems are problems are problems. The problems are problems are problems are problems are problems are problems. The problems are problems are problems are problems are problems are problems. The problems are problems are problems are problems are problems are problems. The problems are problems are problems are problems are problems are pr	:hasMetric puml:arrow puml:up.	<pre><person (customer_id)!hasaddress!address=""></person></pre>	e rr:TriplesMap :	a rr:TriplesMap :	1:
restembly cperson/(ustomer_id)/address' restembly restall  respective this address are stated by the state of	<pre><person (customer="" id)=""> puml:right <person (customer="" id)="" marita=""></person></person></pre>	a rr:FredicateObjectNap;	rr:logicalTable	rr:logicalTable cperson/(customer_id)/churnLikeliho	rr:predicate :value .
restantly restan			rr:sublectMap cperson/(customer id)!subi) .	rr:subjectMap (person/(customer_ld)/churnLikelihi	<pre>cperson/(customer ld)/fraudScore!ltable&gt;</pre>
respectate instables.  **person/(costome_alp)thstatilesals**  **person/(costome_alp)thstatilesals**  **predicateObjectMpg   recibjectMpg   re		rr:termType rr:IRI			rr:tableName "_final.customer".
cperson/(customer_id)thatfaultenails		13	<pre><person (customer_id)!religion!(religion)=""></person></pre>	<pre>(person/(customer_id)/churnL1kel1hood!subj&gt;</pre>	The state of the s
<pre>cperson/(customer_id)/hostFaullhealls</pre>		rr:presicate :nasappress .	recoblectMap [ a recoblectMap :	rr:class :Churntikelihood :	a cr:TriplesMap :
a rribredicateOjectNp; rribredicateOjectNp; rricerally rribredicateOjectNp; rricerally rribredicateOjectNp; rricerally rribredicateOjectNp; rribredicateOjec		<pre>cperson/(customer_id)thasEmailtemail&gt;</pre>	rr:template "{religion}";		rr:logicalTable (person/(customer_id)/fraudScc)
rettemplate "person/(customer_ld)/enall"; rettemplate "rettligion.  rettemplate "rettligion.  rettemplate "rettligion.  rettemplate "rettligion.  rettligion.  re		a rr:PredicateObjectMap ;	rr:ternType rr:Literal		rr:predicateObjectMap cperson/(customer_id)/fraudScr
rriterallyse rriEI  (person/(cuttomer_id))subj.  rritgle(Hup; rritgle(		rr:template "person/(customer id)/email" :	rr:predicate :religion .	a pr:PredicateObjectMap :	rrisubjectMap cperson/(customer_ld)/fraudScc
composition		rr:termType rr:IRI		rr:objectNap ( a rr:ObjectNap ;	<pre><person (customer_ld)="" fraudscore!subj=""></person></pre>
restant introduction of the company		11	<pre>cperson/(customer_id)!subj&gt;</pre>	rridatatype xsd:double ;	a rn:SubjectNap :
coerson/(customer ld)!hasEvent!1)			rr:class :NaturalPerson :	rr:template "(churn_likelihood)";	rritemplate "person/(customer id)/fraudicore"
		<pre>cperson/(customer_ld)!hasEvent!1&gt;</pre>	rr:template "person/(customer_ld)" .	11	
a rr:Predicate(DjectMap; rr:predicate value . cperson/(customer_ld)/fraudscore/value/(fraud_score)>		a rr:PredicateObjectMap;		rn:predicate :value .	<pre>cperson/(customer_ld)/fraudScore!value!(fraud_score)&gt;</pre>



# **Backup slides**

Will present them if there is time





#### Semantic ETL Desiderata

Main requirements for semantic ETL (green: yes or positive, orange: maybe or warning, red: not or negative)



## ETL and Modeling

- Declarative approach: conversion scripts or queries should be written in a high-level language (such as SPARQL, XSPARQL, YARRML, LIXR). This ensures correctness and easier maintainability.
- Generation: conversion scripts should be generated from semantic models, which are also used to generate validation scripts. This saves further effort, ensures correctness and easier maintainability.
- Polyglot Modeling: use the same set of models to generate a variety of technical artefacts, ensuring they are consistent and up to date: conversions, validations, JSON-LD contexts and frames, UI forms, etc.



#### **ETL Performance**

#### Streaming/chunking:

- Tools that collect RDF output to a memory RDF model and then output it only work for thousands of records.
- To work for millions records, the tool must output each piece of RDF as soon as it's ready, otherwise memory requirements and execution time grow dramatically. OntoRefine and TARQL do.
- One pass: ETL tools should read the input data only once.
  - OntoRefine does (on simple tabular data).
  - R2RML mappers should process all maps applying to the same query/table at once for each row, instead of iterating over the same resultset/table multiple times.

#### Direct update:

- ETL tool should be able to output directly to a semantic repository (e.g. GraphDB), both for ingest and update. OntoRefine does, TARQL doesn't.
- This also allows to join to existing semantic data, e.g. to resolve keywords to nomenclature nodes.



### **ETL Logistics**

- SRGO Methodology
  - Steps, Reasoning, Repos/Graphs, Ordering, Ontologies, Reasoning.
  - Strictly describe the What, Where, When, How of a pipeline.
- Prerequisite for:
  - Making the pipeline reproducible
  - Making the KG building process sustainable in the long run
- What: are the steps of the pipeline process.
  - RDF (usually Turtle): obtained from LOD, written by hand (small), converted once (static)
  - Non-RDF data: where to get them from, what credentials are needed
  - Ingestion scripts (declarative or procedural)
  - Virtualization transforms: ONTOP or R2RML mappings; MongoDB JSON connector
  - Transformation/correction: SPARQL Updates that can transform data, derive new data, add missing data, correct fields
  - Validation steps: SHACL, SHEX or SPARQL validations



## ETL Logistics (2)

- Where the steps take place:
  - Several Repositories for data separation or independent update cycles
    - GraphDB Internal Federation eliminates network overhead and improves security
  - Named Graphs for:
    - Unit of Work
    - Idempotence
    - Unit of Transfer
    - Provenance
- When each step is executed:
  - The graph nature of RDF and GraphDB's incremental inference ensure that RDF steps are compositional, i.e. can be done in any order
  - Other steps have ordering requirements, which are usually sparse. This partial order needs to be resolved to a total order for step execution
- **How**: what inference is needed by the pipeline
  - Reasoning ruleset: RDFS, RDFS+, OWL-Horst, OWL RL, OWL QL, etc
  - Ontologies loaded. Many ontologies come with heavy inference (part of heavy "ontological commitment") that you may not need



### ETL Predictability and Transactionality

#### Locality:

- Each ETL step should operate in its own graph and should not disturb any data that is the responsibility of other steps.
- Depends on ETL logistics and using named graphs

#### Predictability and Idempotence:

- The results of each step should not depend on the previous state of its own graph (but can depend on the data of previous steps), in order to be predictable and reproducible.
- Running the same step twice should produce the same result (idempotence).
   Both satisfied by OntoRefine with UPDATE queries.

#### Transactionality:

- Each step should leave the total KG operational and with full data, no matter how long it takes.
- ETL should use transactional (ACID) semantics or use a staging data store, in order to avoid prolonged down-time of the KG.



#### ETL Virtualization and Automation

- NoETL: if source data is very large (Volume) or changes very frequently (Velocity), it may be better to use data in its original form and provide SPARQL querying over it, rather than converting it to RDF
  - Relational Virtualization using ONTOP
  - JSON "virtualization" using <u>MongoDB Integration</u>
- Orchestration: for complex ETL pipelines or when one needs to track multiple ETL instances
  - Use orchestration tools like CWL, Airflow, Apache HOP, etc.
- Automation: to ensure reproducibility in different developer and production environments, package up complex steps and whole pipelines
  - Using automated deployment tools like Docker, Kubernetes, Helm.

