



# SQL to SparkQL

Continue the Development of SQL to SPARQL translation on Jyphon

University of Texas at Austin  
CS347 - Database Management  
Dr. Philip Cannata (Oracle Engineer)  
Nelma Perera Rodriguez, Semur Nabiev, Marek Bejda

# Building Instructions

1. ant clean
2. move dist to the trash
3. ant
4. ant
5. sudo dist/bin/jython

```
>> conn = connectTo 'jdbc:oracle:thin:@129.152.144.84  
/PDB1.usuniversi01134.oraclecloud.internal'  
'cs370_ontologies' 'orcl' 'rdf_mode' 'emp';  
>> SQL on conn 'select * from emp'
```

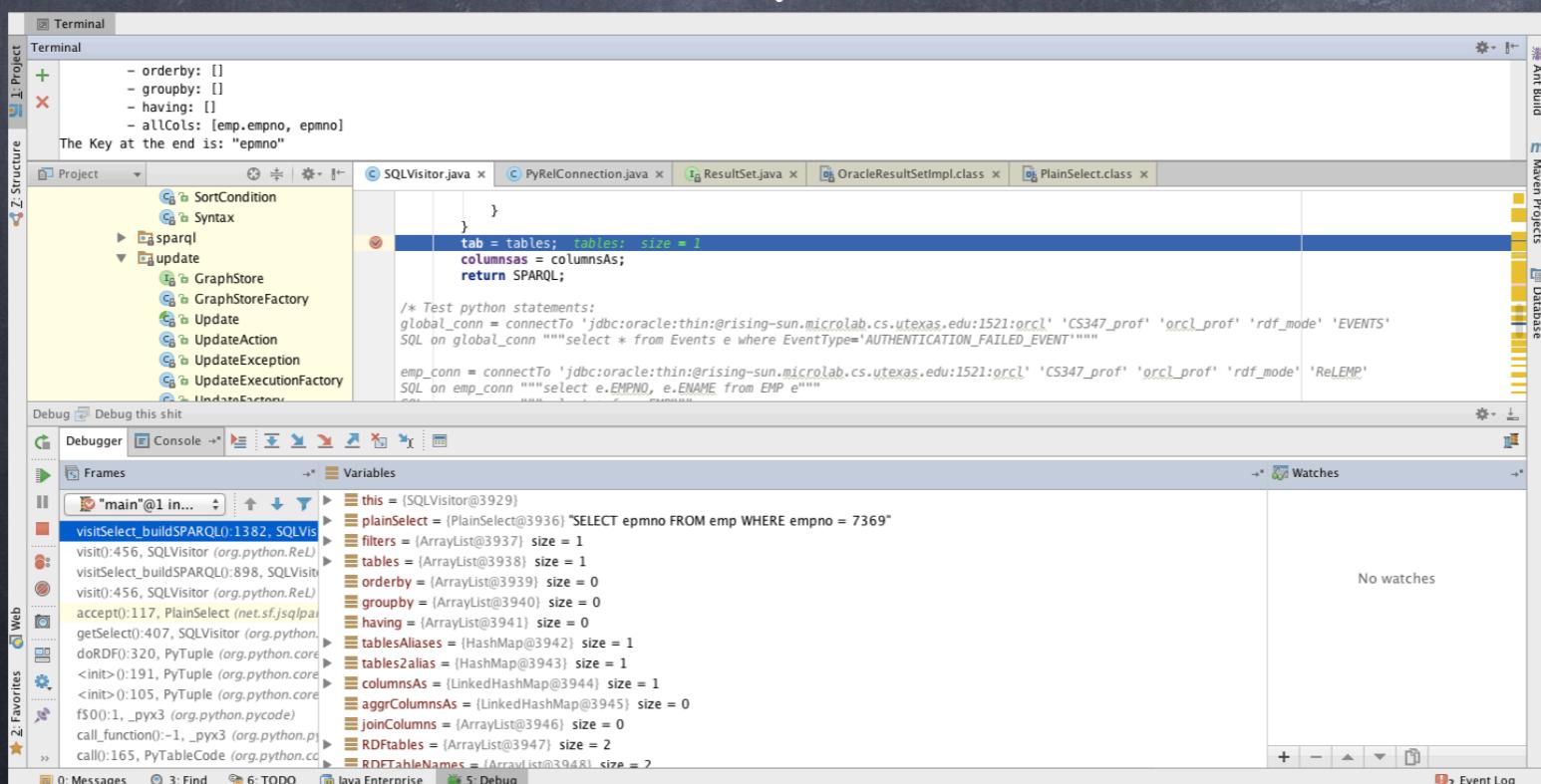
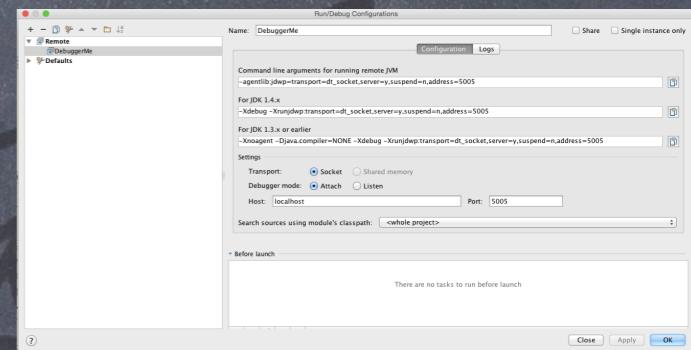


# Debugging Instructions



# Debugging Instructions

1. `export JAVA_OPTS="-agentlib:jdwp=transport=dt_socket,server=y,suspend=n,address=5005"`
2. IntelliJ IDEA (IDE) -> Remote Configuration Profile
3. ant
4. dist/bin/jython
5. `>> conn = connectTo 'jdbc:oracle:thin:@129.152.144.84 /PDB1.usuniversi01134.oraclecloud.internal' 'cs370_ontologies' 'orcl' 'rdf_mode' 'emp';`
6. Set breakpoint in IDE
7. `>> SQL on conn 'select * from emp'`



# Debugging Instructions

The screenshot shows an IDE interface with the following components:

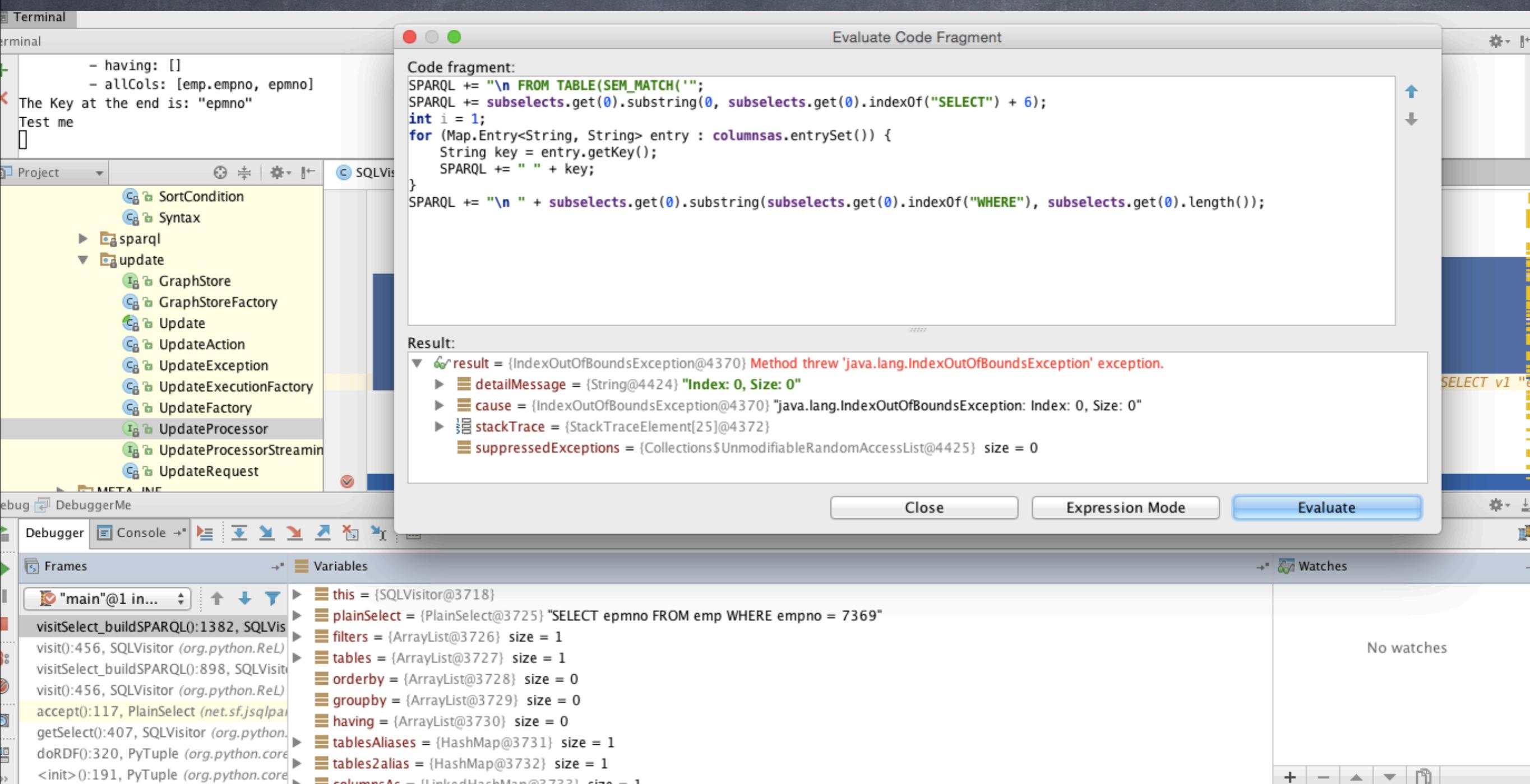
- Terminal:** Displays the output of a command-line session:

```
- orderby: []
- groupby: []
- having: []
- allCols: [emp.empno, empno]
The Key at the end is: "empno"
```
- Project:** Shows a tree view of project files under the `update` package, including `GraphStore`, `GraphStoreFactory`, `Update`, `UpdateAction`, `UpdateException`, `UpdateExecutionFactory`, and `UpdateFactory`.
- Code Editor:** Shows a Java file (`SQLVisitor.java`) with code related to SPARQL generation. A breakpoint is set on the line `tab = tables; tables: size = 1`. The code also includes test Python statements for connecting to Oracle databases and executing SQL queries.
- Debugger:** Shows the current stack trace:

```
"main"@1 in... visitSelect_buildSPARQL():1382, SQLVis
```

and a list of variables:
  - `this = {SQLVisitor@3929}`
  - `plainSelect = {PlainSelect@3936} "SELECT empno FROM emp WHERE empno = 7369"`
  - `filters = {ArrayList@3937} size = 1`
  - `tables = {ArrayList@3938} size = 1`
  - `orderby = {ArrayList@3939} size = 0`
  - `groupby = {ArrayList@3940} size = 0`
  - `having = {ArrayList@3941} size = 0`
  - `tablesAliases = {HashMap@3942} size = 1`
  - `tables2alias = {HashMap@3943} size = 1`
  - `columnsAs = {LinkedHashMap@3944} size = 1`
  - `aggrColumnsAs = {LinkedHashMap@3945} size = 0`
  - `joinColumns = {ArrayList@3946} size = 0`
  - `RDFtables = {ArrayList@3947} size = 2`
  - `RDFTableNames = {ArrayList@3948} size = 2`
- Event Log:** Shows the following log entries:
  - 0: Messages
  - 3: Find
  - 6: TODO
  - Java Enterprise
  - 5: Debug
  - Event Log

# Debugging Instructions



# SparQL Executing Instructions

Oracle SQL Developer : cs347\_mb39822

The screenshot shows the Oracle SQL Developer interface. The top menu bar displays the title "Oracle SQL Developer : cs347\_mb39822". The left sidebar has a "Connections" section with one entry: "cs347\_mb39822". Under this connection, there is a "Tables (Filtered)" section listing various tables such as ACTIVE\_INVOICES, ADDRESSES, ADMINISTRATORS, CATEGORIES, COLOR\_SAMPLE, CUST\_JSON, CUSTOMERS, DATE\_SAMPLE, DEPARTMENTS, DEPT, E3\_C##CS347\_MB39822\_DATA, E5\_C##CS347\_MB39822\_DATA, EMP, EMPLOYEES, FLOAT\_SAMPLE, GENERAL\_LEDGER\_ACCOUNTS, INVOICE\_ARCHIVE, INVOICE\_LINE\_ITEMS, INVOICES, ITEMS, NULL\_SAMPLE, ORDER\_DETAILS, ORDER\_ITEMS, ORDERS, PAID\_INVOICES, and PRODUCTS. The main workspace is titled "Worksheet" and contains the following SparQL query:

```
SELECT v1 "ename", v2 "dname"
  FROM TABLE(SEM_MATCH(
    SELECT * WHERE {
      GRAPH <emp_SCHEMA> { ?s1 rdf:type :emp }
      GRAPH <dept_SCHEMA> { ?s2 rdf:type :dept }
      OPTIONAL { ?s1 :ename ?v1 }
      OPTIONAL { ?s2 :dname ?v2 }
      ?s1 :deptno ?j1 .
      ?s2 :deptno ?j1 .
    },
    SEM_MODELS('EMP_CS370_ONTOLOGIES'), null,
    SEM_ALIASES( SEM_ALIAS('', '#')), null ) );
```

Below the worksheet is a "Query Result" window. It shows the SQL command being executed and a series of ORA- errors indicating that the model "EMP\_CS370\_ONTOLOGIES" does not exist. The errors are:

- ORA-55300: model "EMP\_CS370\_ONTOLOGIES" does not exist
- ORA-06512: at "MDSYS.MD", line 1723
- ORA-06512: at "MDSYS.MDERR", line 17
- ORA-06512: at "MDSYS.RDF\_APIS\_INTERNAL", line 1832
- ORA-06512: at "MDSYS.RDF\_MATCH\_IMPL\_T", line 381
- ORA-06512: at line 1

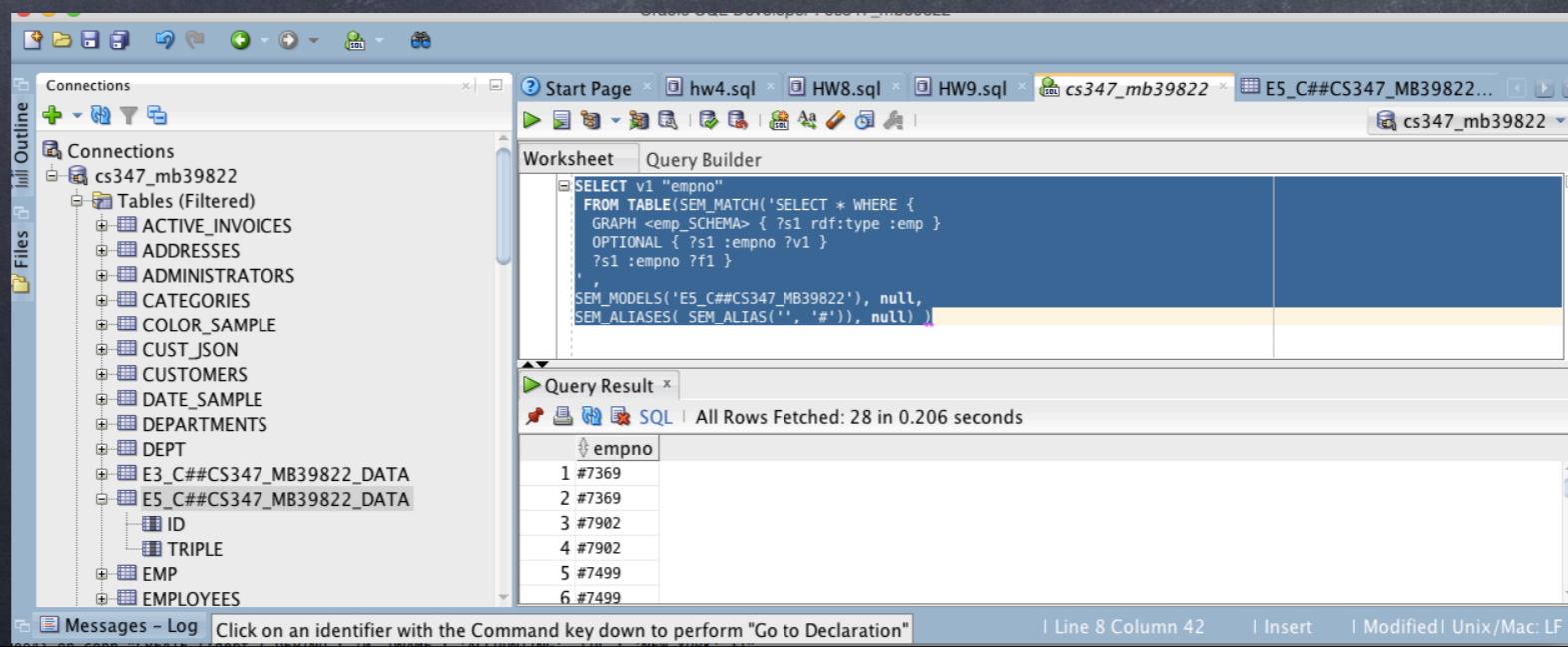
Following the errors, there is a note about the cause and action:

55300.0000 - "model %s does not exist"  
\*Cause: The specified model could not be found.  
\*Action: Make sure that the model has been created.

The bottom left corner shows a "Messages - Log" tab.

# SparQL Executing Instructions

1. Login to a CS machine
2. wget [https://www.dropbox.com/s/c0gshqi3s1fhx9n/neo4j\\_emp\\_db.txt?dl=1](https://www.dropbox.com/s/c0gshqi3s1fhx9n/neo4j_emp_db.txt?dl=1) -O emp\_dept.neo4j.formatted.txt
3. Change the login credentials  
conn = connectTo 'jdbc:oracle:thin:@sayonara.microlab.cs.utexas.edu:1521:orcl' 'C##cs347\_UTEID' 'orcl\_UTEID' 'rdf\_mode' 'E5';
4. /u/cannata/ReL/dist/bin/jython emp\_dept.neo4j.formatted.txt
5. Now open SQL Developer and you should see a table named "E5\_C##cs347\_UTEID"
6. Now you can run the SparQL queries.



# Instructions

G. Implement the conversion of SQL subqueries to SPARQL subqueries in ReL (ReL Source Code).

- ④ From ✓
- ④ Where ✓
- ④ Select
- ④ JOIN

# How it works

SQL Query

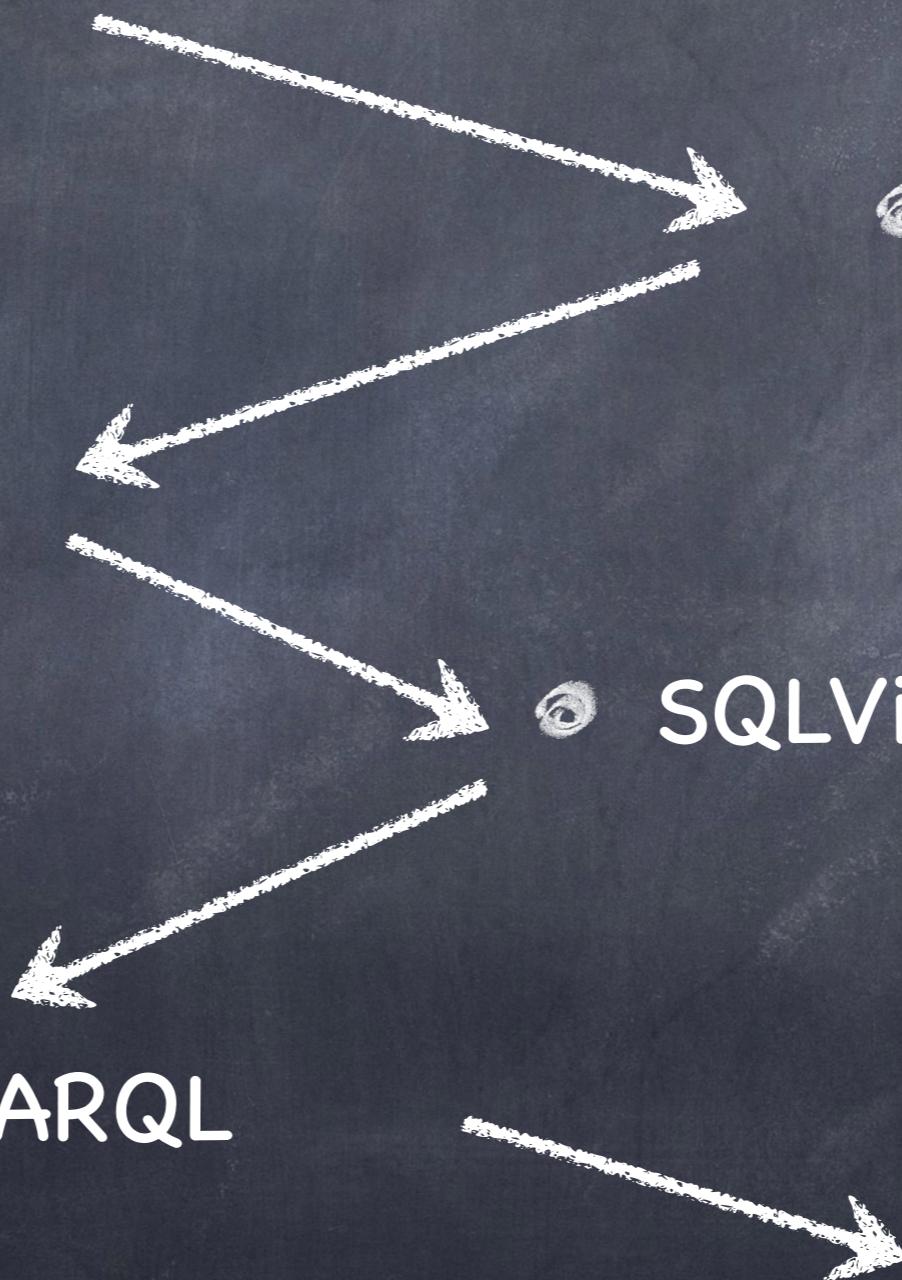
IF SELECT

visitSelect\_buildSPARQL

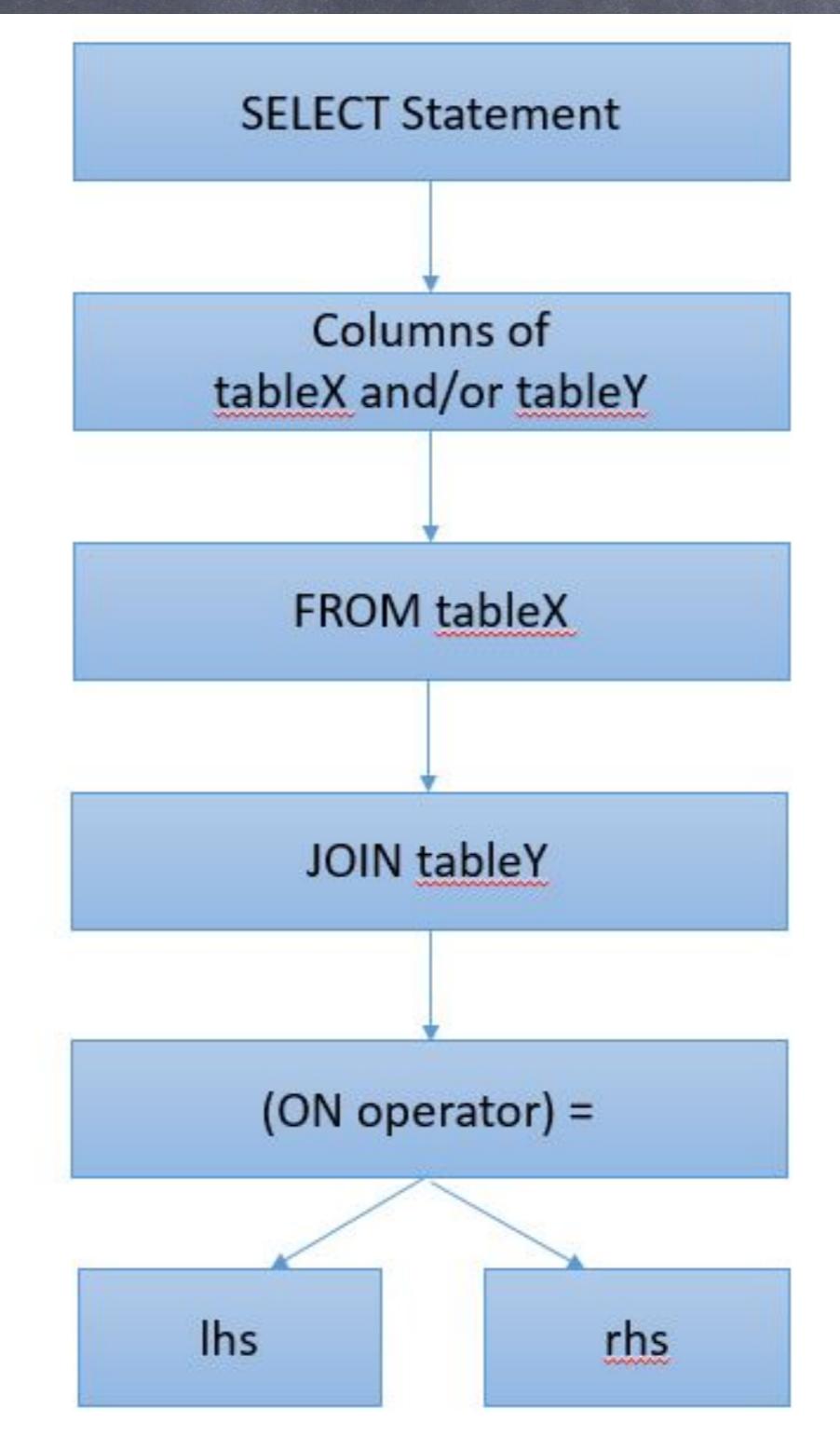
SQL Server

SQLVisitor.getSelect

SPARQL



# old design



# Example without subqueries

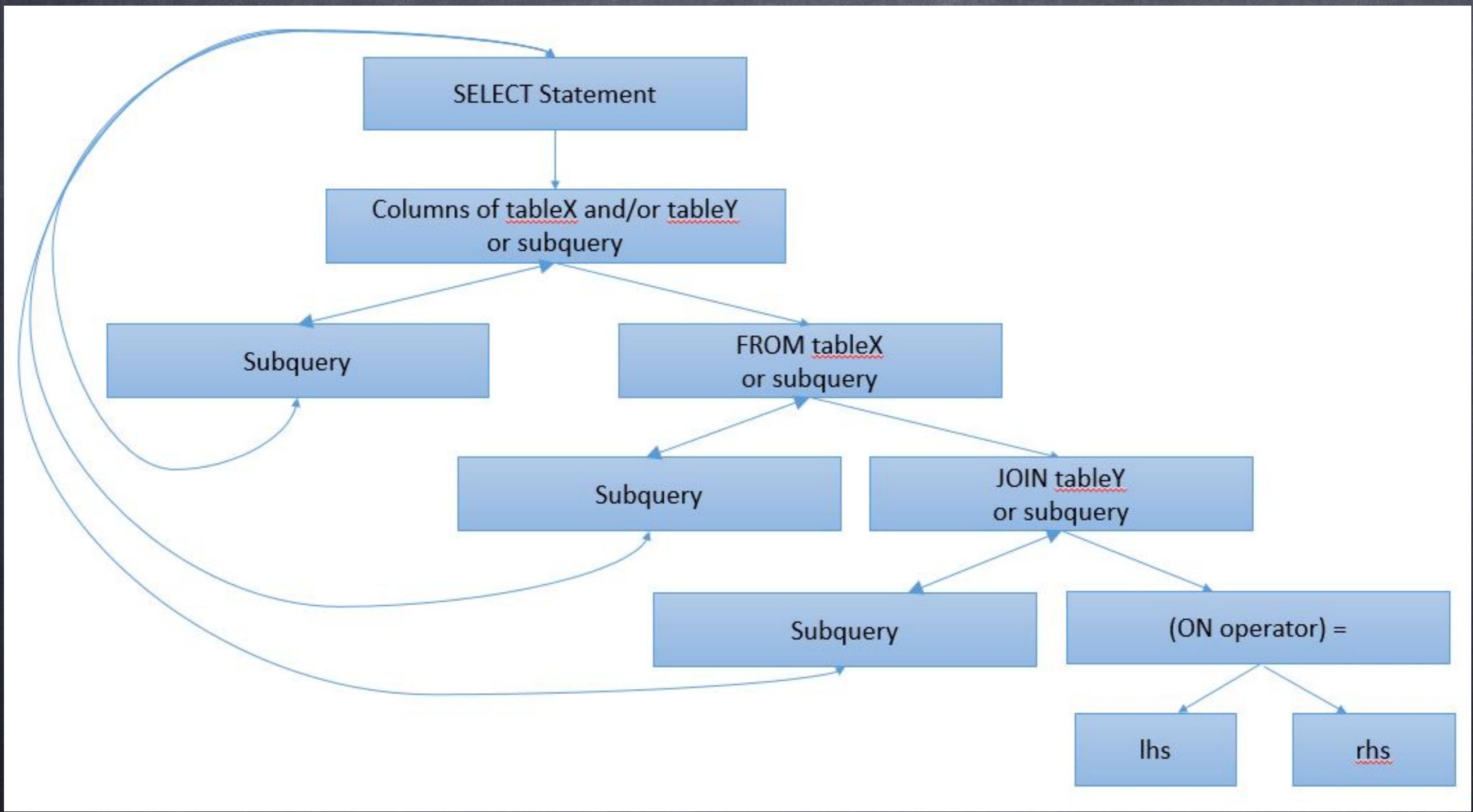
SQL:

```
SELECT * FROM emp;
```

SPARQL:

```
SELECT v1 "empno", v2 "deptno", v3 "comm", v4 "job", v5 "mgr", v6 "ename",
v7 "hiredate", v8 "sal", v9 "dept"
FROM TABLE(SEM_MATCH('SELECT * WHERE {
    GRAPH { ?s1 rdf:type :emp }
    OPTIONAL { ?s1 :empno ?v1 }
    OPTIONAL { ?s1 :deptno ?v2 }
    OPTIONAL { ?s1 :comm ?v3 }
    OPTIONAL { ?s1 :job ?v4 }
    OPTIONAL { ?s1 :mgr ?v5 }
    OPTIONAL { ?s1 :ename ?v6 }
    OPTIONAL { ?s1 :hiredate ?v7 }
    OPTIONAL { ?s1 :sal ?v8 }
    OPTIONAL { ?s1 :dept ?v9 } })
```

# New design



# Code Changes

- Create the List data structure subselects to store a list af all subqueries statements in the query, with size = to subDepth, and it will be filled depending on how many trues are in sq stack
- boolean subSelect will tell us if we processing a query of subquery inside visit(PlainSelect plainSelect) and visitSelect\_buildSPARQL
- subDepth will tell us how many subqueries there are in the query
- Pass a list as an arg to visitSelect\_buildSPARQL method
- Then in visitSelect\_buildSPARQL we get the subquery from the SQL FROM OR WHERE statement then we modified the SPARQL FILTER statement to form the translations.

# Subquery in WHERE

SQL:

```
SELECT empno FROM emp  
WHERE empno=(SELECT empno FROM emp WHERE empno=7369);
```

SPARQL:

```
SELECT v1 "empno"  
FROM TABLE(SEM_MATCH('SELECT * WHERE {  
    GRAPH <emp_SCHEMA> { ?s1 rdf:type :emp }  
    OPTIONAL { ?s1 :empno ?v1 }  
    {  
        SELECT * WHERE {  
            GRAPH <emp_SCHEMA> { ?s1 rdf:type :emp }  
            OPTIONAL { ?s1 :empno ?v1 }  
            ?s1 :empno ?f1 .  
            FILTER(?f1 = :7369)}  
    }'))
```

RESULTS: (('empno',), (7369,), (7369,))

# Subquery with \*

SQL:

```
SELECT * FROM emp  
WHERE empno=(SELECT epmno FROM emp WHERE empno=7369);
```

SPARQL:

```
SELECT v1 "empno" v2 "" v3...  
FROM TABLE(SEM_MATCH('SELECT v1 WHERE {  
    GRAPH { ?s1 rdf:type :emp }  
    OPTIONAL { ?s1 :empno ?v1 }  
    FILTER( ?v1 = :(SELECT empno FROM emp WHERE epmno=7369) )  
}'))
```

RESULTS:

# Subquery in FROM

SQL:

```
SELECT empno, sal FROM (SELECT empno, dept, sal FROM emp);
```

SPARQL:

```
SELECT v1 "empno", v3 "sal"
FROM TABLE(SEM_MATCH('SELECT ?v1 ?v2 ?v3
WHERE {
    GRAPH <emp_SCHEMA> { ?s1 rdf:type :emp }
    OPTIONAL { ?s1 :empno ?v1 }
    OPTIONAL { ?s1 :dept ?v2 }
    OPTIONAL { ?s1 :sal ?v3 }
}'))
```

RESULTS: (('empno', 'sal'), (7369, 800), (7369, 800), (7369, 800),

# Additional Resources

SQL Nested Queries in SPARQL <http://ceur-ws.org/Vol-619/paper5.pdf>

SubQuery Standard <http://www.w3.org/TR/sparql11-query/#subqueries>

SparQL By Example <http://www.cambridgesemantics.com/semantic-university/sparql-by-example>

Wiki SubSelect <http://www.w3.org/2009/sparql/wiki/Design:SubSelect>

New

