A Simple Interface for Non Standard Knowledge Systems (SINKS)

By

Harini Rao

Advisor: Dr. Christopher Pollett
Objective

- To deploy a deductive database system such as XSB, as a back end for a relational database, Oracle.

- Application programmers can use SQL.
Introduction

- Deductive databases - integration of relational databases and logic programming techniques.

- A deductive database system - a database system that includes capabilities to define deductive rules which can deduce or infer additional information from the facts that are stored in a database.
Advantages

- Combine benefits of two approaches.
- Provide means for expressing negation and disjunction.
- Query processing is much simpler and easier.
Disadvantage

- Most database application programmers are unfamiliar with logic programming.
XSB

- a deductive database system developed at the computer science department, Stony Brook University.

Features:

- Uses goal-directed resolution strategy to solve Prolog’s problems.
- Evaluates stratified queries much faster.
- Tabling - to evaluate programs with negation.
Design

- Oracle–XSB interface is a subsystem that allows Oracle users to access XSB databases.

- Allows facts in XSB to be accessed from Oracle’s environment.

- Permits users to write explicit SQL statements.
Data Flow Diagram

- User Query
  - INSERT or UPDATE or DELETE
  - SELECT
- Oracle Database
- Updatable View
- INSTEAD OF trigger
- Stored Procedure
- C Procedure
- C Procedure
- C-XSE Interface
- XSE Database
- Text File
The Oracle-XSB interface has two sub components.

- View Level Interface - translates SQL queries into Prolog clauses.

- C-XSB interface (provided with XSB) - allows a C program to pass queries to XSB.
View Level Interface

- Translates a complex database query into a combination of one or more Prolog rules.

- Its design includes updatable views, instead of triggers or stored procedures and external C procedures.
Updatable Views

- View - a virtual table whose contents is defined by a query.

- A view is not modifiable if its view query contains joins, set operators, aggregate functions, GROUP BY, CONNECT BY, START WITH clauses or DISTINCT operator.
Instead of Triggers

- Instead of trigger - tells Oracle how to process a DML operation performed on a view.

- Execute the body of trigger instead of performing the actions that invoked the trigger.
Stored Procedures

- Stored procedure - Named group of SQL statements previously created and stored in the server database.

- Reduce network traffic.

- Improve performance and security.
External Routines

- External routine - a third-generation language procedure stored in a DLL, and called by the DBA to perform special-purpose processing.

- PL/SQL calls the routine as if it were a PL/SQL subprogram.
C - XSB Interface

Calling XSB from C:

- Several functions - that allow a C program to initialize and interact with XSB.

- They pass commands or queries to the XSB system.
C-Callable Functions

Some of these functions are:

- `int xsb_init(int argc, char *argv[])`: Used for initializing XSB.
- `int xsb_command()`: Passes a command to XSB.
- `int xsb_query()`: Used for passing a query to XSB.
- `int xsb_next()`: Returns answers to the calling program if the query has multiple data answers.
- `int xsb_close()`: Completely closes the connection and no other calls can be made to XSB.
Deployment

- Combine queries from different database systems.
- Retrieve data from either XSB or Oracle or from both.
Example - Retrieving Data only from XSB

Assume the table declarations:
emp(ename,job,sal,comm,deptno).
depth(deptno,dname,loc).

the SQL statement:
SELECT empno,comm,hiredate,dname from dept,emp;

generates the Prolog query,
emp(EMPNO,_,_,_,HIREDATE,_,COMM,_),dept(_,DNAME,_).
and the results:

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>7934</td>
<td>565</td>
<td>1-jun-1980</td>
<td>research</td>
<td></td>
</tr>
<tr>
<td>7934</td>
<td>565</td>
<td>1-jun-1980</td>
<td>sales</td>
<td></td>
</tr>
<tr>
<td>7934</td>
<td>565</td>
<td>1-jun-1980</td>
<td>operations</td>
<td></td>
</tr>
<tr>
<td>7834</td>
<td>_</td>
<td>15-jan-1985</td>
<td>research</td>
<td></td>
</tr>
<tr>
<td>7834</td>
<td>_</td>
<td>15-jan-1985</td>
<td>sales</td>
<td></td>
</tr>
<tr>
<td>7834</td>
<td>_</td>
<td>15-jan-1985</td>
<td>operations</td>
<td></td>
</tr>
<tr>
<td>7782</td>
<td>_</td>
<td>9-jun-1981</td>
<td>research</td>
<td></td>
</tr>
<tr>
<td>7782</td>
<td>_</td>
<td>9-jun-1981</td>
<td>sales</td>
<td></td>
</tr>
<tr>
<td>7782</td>
<td>_</td>
<td>9-jun-1981</td>
<td>operations</td>
<td></td>
</tr>
</tbody>
</table>
A more complicated example

The SQL statement:
SELECT dname, empno, mgr, hiredate, FROM dept, emp, where deptno=20');

generates the rule,
equalto(DNAME,EMPNO,MGR,HIREDATE) :-
dept(X,DNAME,_), emp(EMPNO,_,_ ,MGR,HIREDATE,_,_,X), X = 20.
and the results:

Answer

*******************************************************************************
research 7844 _ _ _ 
research 7369 7902 17-dec-1980
research 7566 7839 2-apr-1981
*******************************************************************************
Example - Retrieving Data from both XSB and Oracle

To retrieve the data from both Oracle and XSB, invoke a stored procedure as follows:

```
begin
    Select_From('SELECT empno,colmpos,colmname FROM emp,sampletable');
end;
```

Equivalent XSB query generated:
```
emp(EMPNO,_,_,_,_,_,_,_).
```
Results

Answer

*********************************
7934   roy
7844   keeth
7780   russell
7834   george
*********************************

Oracle Results

*********************************
1      col1
2      col2
3      col3
4      col4
*********************************
Insertions and deletions

Insertion:
To insert a new row to a table, invoke an instead of trigger as follows:

```sql
begin
  Insert_into('emp', '#empno#ename#job#mgr#
              hiredate#sal#comm#deptno#');
end;
```
The SQL statement,

```
INSERT INTO emp VALUES(7390,'KIT','CFO',7342,'10-OCT-1960',56344,0,20);
```

adds a new fact to the database as

```
emp(7390,kit,cfo,7342,10-oct-60,56344,0,20).
```
Deletion

To delete a fact from a table based on a condition:

begin
  Delete_from('emp','sal');
end;

The SQL statement,
DELETE FROM emp WHERE sal = 1680;
deletes the fact from the emp table.
Applications

- Many applications - in the fields of biotechnology and genetic research.
- Use - potential replacement to access data, previously stored using legacy systems.
- Limit the scope of queries and potentially improve the speed of their evaluation.
Sample application scenario

- Presence of two different database systems – provides access to both separately and simultaneously.

- Users and application programmers can access knowledge based systems use SQL.

- Reduces time and effort.
Challenges

- Translation of SQL queries into prolog queries.
- Retrieval of data without updating it.
- Accessing external C procedures from Oracle.
- Calling XSB from C.
- Returning the results from XSB to Oracle.
Future Enhancements

- A faster implementation of the interface reducing the data retrieval time.
- Creating a virtual drive and allocating some memory in RAM.
- Using Java external procedures.
Conclusion

- Completely different from the XSB-Oracle interface.
- Uses a novel approach to access XSB from Oracle.
- Allows the users to retrieve facts (in XSB), from Oracle’s environment as though they existed as tables.
Conclusion (cont.)

-Permits the users to write SQL statements to access data from XSB.

-Does not require the user to be familiar with logic programming.

-Provides a method to perform updates on the logical databases.
Questions