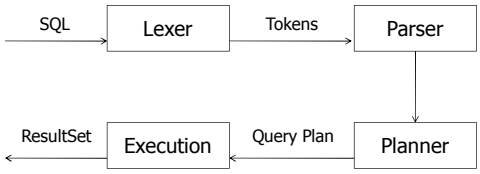
CS422 Principles of Database Systems

Introduction to Query Processing

Chengyu Sun
California State University, Los Angeles

Query Processing in SimpleDB



Schema

- Departments(did, dname)
- Students(sid, sname, dept)

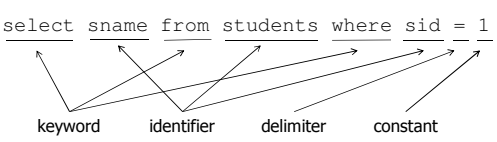
SQL

```
create table departments (
        did             int;
        dname           varchar(10)
);

select sname, dname
        from students, departments
        where dept = did and sid = 1;
```

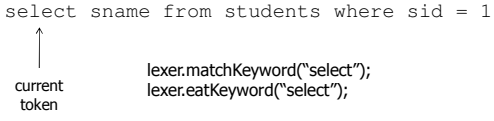
Lexical Analysis

- Split the input string into a series of tokens



Lexer API

- Iterate through the tokens
  - Check the current token – "Match"
  - Consume the current token – "Eat"



Grammar …

```
<Field>      := IdTok
<Constant>   := StrTok | IntTok
<Expression> := <Field> | <Constant>
<Term>       := <Expression> = <Expression>
<Predicate>  := <Term> [ AND <Predicate> ]
```

… Grammar

```
<Query>     := SELECT <SelectList> FROM <TableList>
               [ WHERE <Predicate> ]
<SelectList> := <Field> [ , <SelectList>]
<TableList>  := IdTok [ , <TableList> ]

<CreateTable> := CREATE TABLE IdTok ( <FieldDefs> )
<FieldDefs>   := <FieldDef> [ , <FieldDefs> ]
<FieldDef>    := IdTok <TypeDef>
<TypeDef>     := INT | VARCHAR ( IntTok )
```

From Grammar to Code …

```
public QueryData query()
{
   lex.eatKeyword( "select" );
   Collection<String> fields = selectList();
   lex.eatKeyword( "from" );
   Collection<String> tables = tableList();
   Predicate pred = new Predicate();
   if( lex.matchKeyword("where") )
   {
      lex.eatKeyword("where");
      pred = predicate();
   }
   return new QueryData( fields, tables, pred );
}
```

… From Grammar to Code

```
public Collection<String> selectList()
{
   Collection<String> L = new ArrayList<String>();
   L.add( field() );
   if( lex.matchDelim(',') )
   {
      lex.eatDelim(',');
      L.addAll( selectList() );
   }
   return L;
}

public String field() { return lex.eatId(); }
```

Create Table

- Input: table name and Schema
- Create a record file for the table
- Insert the table information into system catalog

System Catalog

- A.K.A. data catalog, data dictionary
- A set of *tables* containing metadata about the schema elements and data statistics
  - Table, field, view, index information
  - Data statistics
    - E.g. total number of rows in a table and distinct values in a column
    - Used for query optimization

System Catalog Example

- `tblcat` and `fldcat` in SimpleDB

Query Planning

- Break a query into individual operations, and organize them into certain order (i.e. a query plan).

Relational Algebra Operations

- Selection, projection, product
- Join
- Rename
- Set operations: union, intersection, difference
- Extended Relation Algebra operations
  - Duplicate elimination
  - Sorting
  - Extended projection, outer join
  - Aggregation and grouping

Implement Selection

Input

sid	sname
1	Joe
2	Amy

sid=1 →

Output

sid	sname
1	Joe

Implement Projection

Input

sid	sname
1	Joe
2	Amy

sname →

Output

sname
Joe
Amy

Implement Product

Input

sid	sname	dept
1	Joe	10
2	Amy	20

did	dname
10	CS
20	Math

⇩

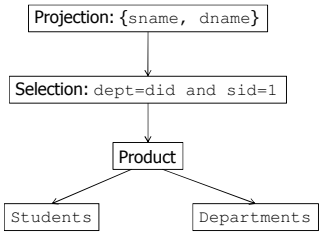
Output

sid	sname	dept	did	dname
1	Joe	10	10	CS
1	Joe	10	20	Math
2	Amy	20	10	CS
2	Amy	20	20	Math

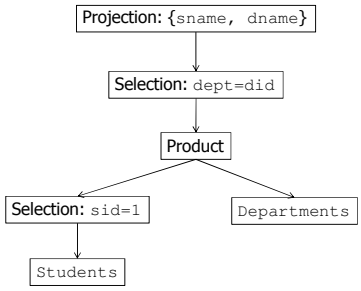
About Implementations of RA Operations

- Each RA operation can be implemented and optimized independently from others
- A RA operation may have multiple implementations
  - E.g. *table scan* vs. *index scan* for selection
- The efficiency of an implementation depends on the characteristics of the data
  - E.g. *nested loop join* vs. *hash join*

A Simple Query Plan



A Better Query Plan – Query Optimization



Query Execution – Scan

- A scan is an interface to a RA operation implementation

```
public interface Scan {

    public boolean next(); // move to the next result
    public int getInt( String fieldName );
    public String getString( String fieldName );

}
```

Scan Example: SelectScan

```
public SelectScan( Scan s, Predicate pred )
{
    this.s = s;
    this.pred = pred;
}

public boolean next()
{
    while( s.next() )
        if( pred.isSatisfied(s) ) return true;
    return false;
}
```

Readings

- Textbook Chapter 16, 17, 18, 19