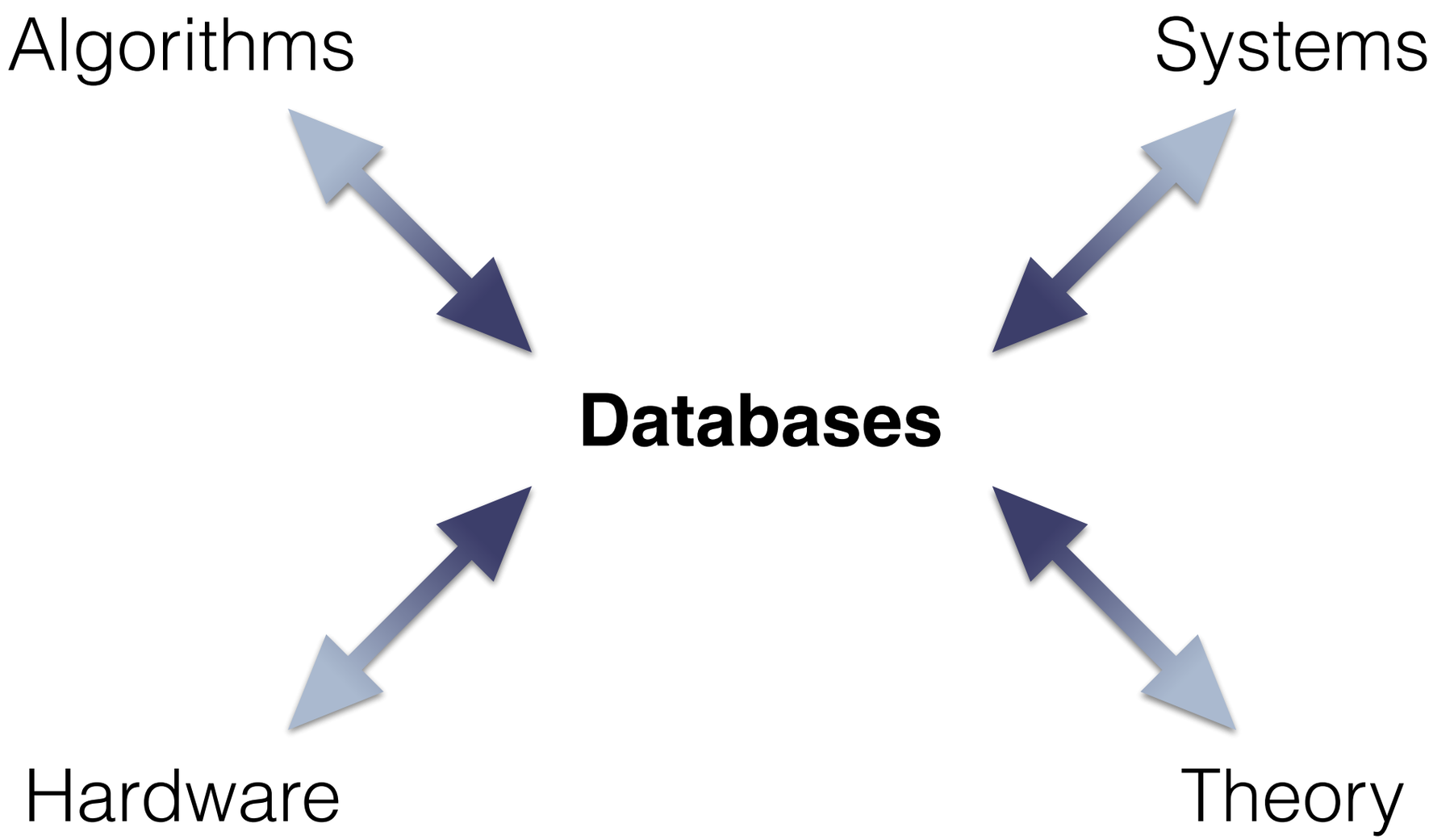
CSE 462 - Databases

Oliver Kennedy
okennedy@buffalo.edu

Why Study Databases?

2 Queries per Second

Interesting Problems



$$$

Rank	Company	Country	Sales	Profits	Assets	Market Value
32	Microsoft	United States	$83.3 B	$22.8 B	$153.5 B	$343.8 B
94	Oracle	United States	$37.9 B	$11.1 B	$86.6 B	$185 B
207	SAP	Germany	$22.3 B	$4.4 B	$37.3 B	$97.1 B
784	VMware	United States	$5.2 B	$1 B	$12.3 B	$48.2 B
848	Symantec	United States	$6.8 B	$0.9 B	$13.3 B	$14 B
998	CA	United States	$4.6 B	$1 B	$11.8 B	$14.1 B
1126	Fiserv	United States	$4.8 B	$0.7 B	$9.7 B	$14.6 B
1153	HCL Technologies	India	$4.7 B	$0.7 B	$4.2 B	$16.6 B
1158	Intuit	United States	$4.2 B	$0.7 B	$4.7 B	$22.4 B
1173	Amadeus IT Holdings	Spain	$4.1 B	$0.7 B	$7.5 B	$18.9 B

8 of the **top 10**
Forbes Global 2000
Software & Programming
Companies

base their business on
data management

What is “Databases”?

Databases

- **How do we ask and answer questions about data?**

- **How do we manipulate and persist data?**

Databases

- **How do we ask and answer questions about data?**

efficiency

accuracy

multiple sources

summaries

- **How do we manipulate and persist data?**

consistency

correctness

parallelism

Database Tools

Techniques:

Data Modeling

Cost-Based Optimization

Recipes:

Join Algorithms

Index Datastructures

Knowledge:

The Memory Hierarchy

Data Consistency

Which tools do you use … and when?

This Course in a Nutshell

This Course in a Nutshell

There might be many correct options…

This Course in a Nutshell

There might be many correct options…
…but some are better than others…

This Course in a Nutshell

There might be many correct options…
…but some are better than others…
…for specific tasks.

This Course in a Nutshell

There might be many correct options…
…but some are better than others…
…for specific tasks.

How do you define ‘correct’ and ‘better’?

This Course in a Nutshell

There might be many correct options…
…but some are better than others…
…for specific tasks.

How do you define ‘correct’ and ‘better’?

How do you find alternatives that are correct?

This Course in a Nutshell

There might be many correct options…
…but some are better than others…
…for specific tasks.

How do you define ‘correct’ and ‘better’?

How do you find alternatives that are correct?

How do you find alternatives that are better?

What is ‘Better’?

- **Declarative Queries**: ‘Easy to think about’ vs ‘Fast’
- **Data Layouts**: Space vs Fast Updates vs Fast Queries
- **Parallel Updates**: Reactive vs Proactive Concurrency

Today

- **Logistics**: What you need to know
- **Project Outline**: Build the next big data startup
- **Ways to Fail**: What not to do and why
- **Intro**: So what is a database anyway?

General Course Information

People

- **Oliver Kennedy** (okennedy@buffalo.edu)
- **Jun Chu** (jchu6@buffalo.edu)
- **Nikhil Londhe** (support role only)

Syllabus & Website

http://odin.cse.buffalo.edu/teaching/cse-462

Course Forum: Piazza

Course Project: DμBStep

Course Structure

- **Programming Assignment** (50% of overall grade)
  - 4-Person Groups
  - Build a relational query engine
- **Course Content** (50% of overall grade)
  - **2 Midterm Exams** (5 or 10% of overall grade each)
  - **Comprehensive Final Exam** (20, 25, or 30% of overall grade)
    - Final Grade replaces up to 5% of each midterm's grade
  - **Homeworks** due on Thursdays (10% of overall grade; drop lowest 2)

Data μBases Step-by Step

(a.k.a., how to be the next ‘big’ data startup)

Embedded Databases

- SQLite (in your browser, computer, phone, etc…)
- Simple, easy-to-use, declarative data management
- Critical for future tech: Part of Mobile, IoT, Web

Embedded Databases

- SQLite (in your browser, computer, phone, etc…)
- Simple, easy-to-use, declarative data management
- Critical for future tech: Part of Mobile, IoT, Web

Your startup's goal…

…build (part of) an embedded database

Data μBases (Step-by Step)

I give you data (CSV Files + Schema)

I ask you a question about the data (SQL)

You give me an answer

Data μBases (Step-by Step)

Real World Challenge: You start with…

… an empty GIT repository

… open-source libraries (more on this next week)

Data μBases (Step-by Step)

Real World Challenge: You get graded on your code's…

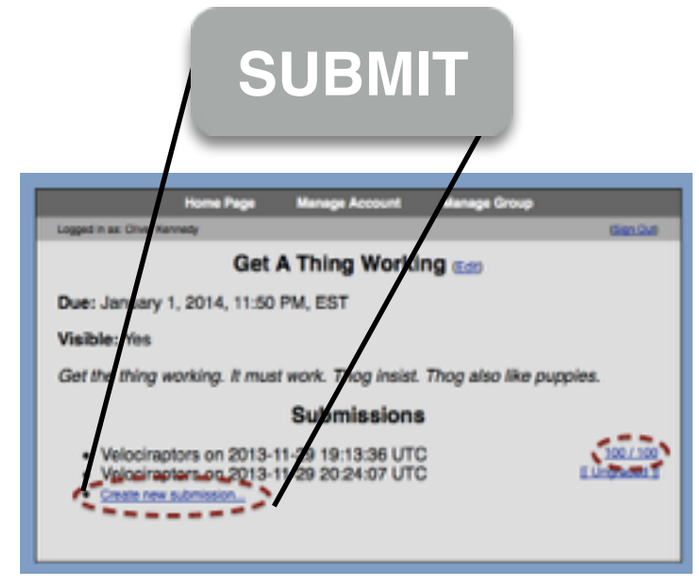
… **correctness** (do you produce the right answer)
minimum 1/3 of grade for producing the right answer

… **speed** (how fast did you produce the answer)
+2/3 for meeting/beating the reference implementation

DμBStep

You write code

You push to GIT



DμBStep compiles your code

π-graders run your code

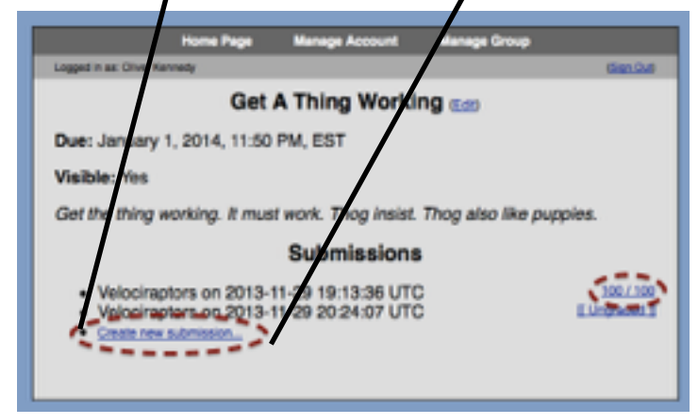
DμBStep emails your group

DμBStep

You write code

You push to GIT

SUBMIT

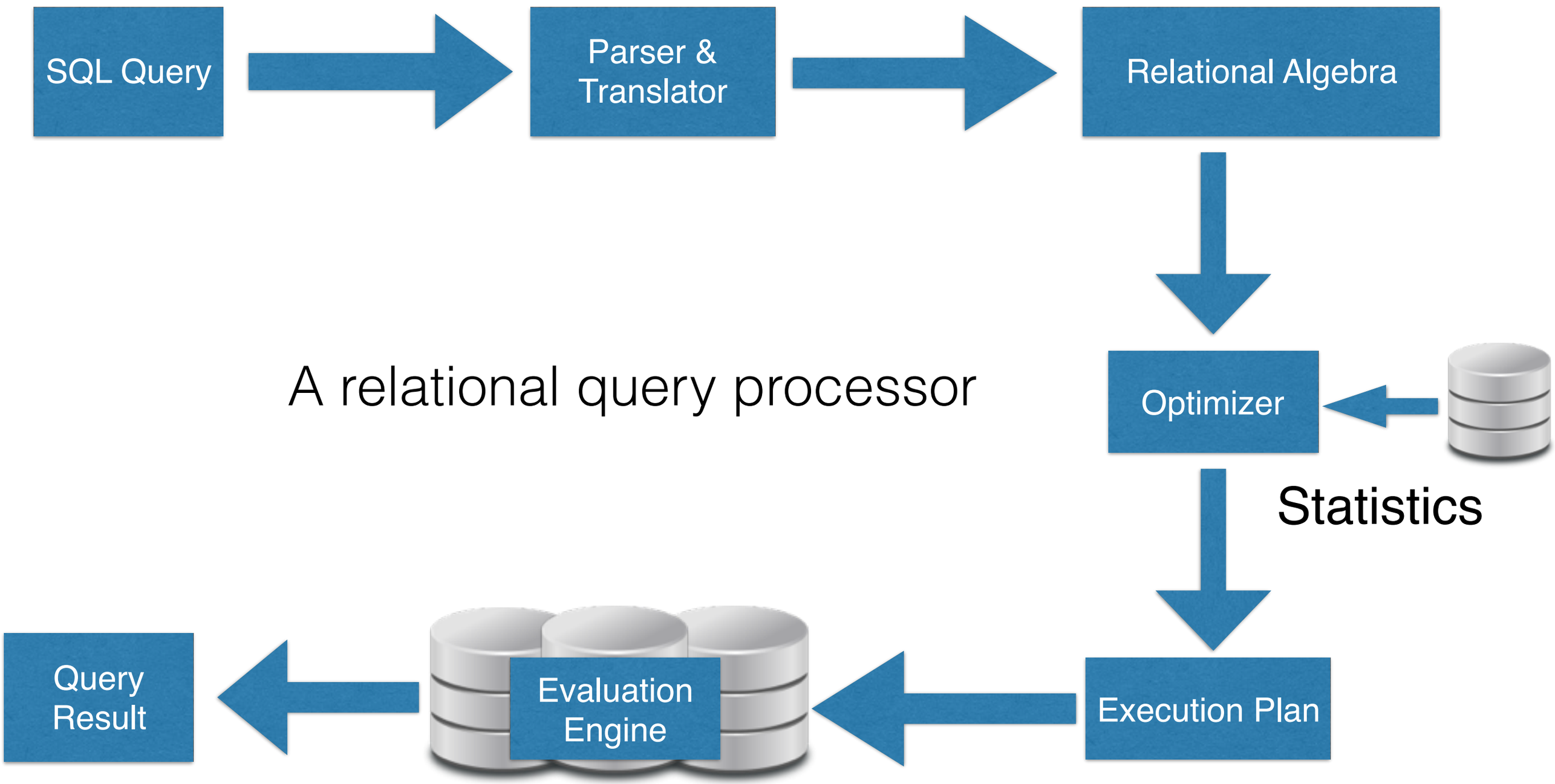


DμBStep compiles your code

π-graders run your code

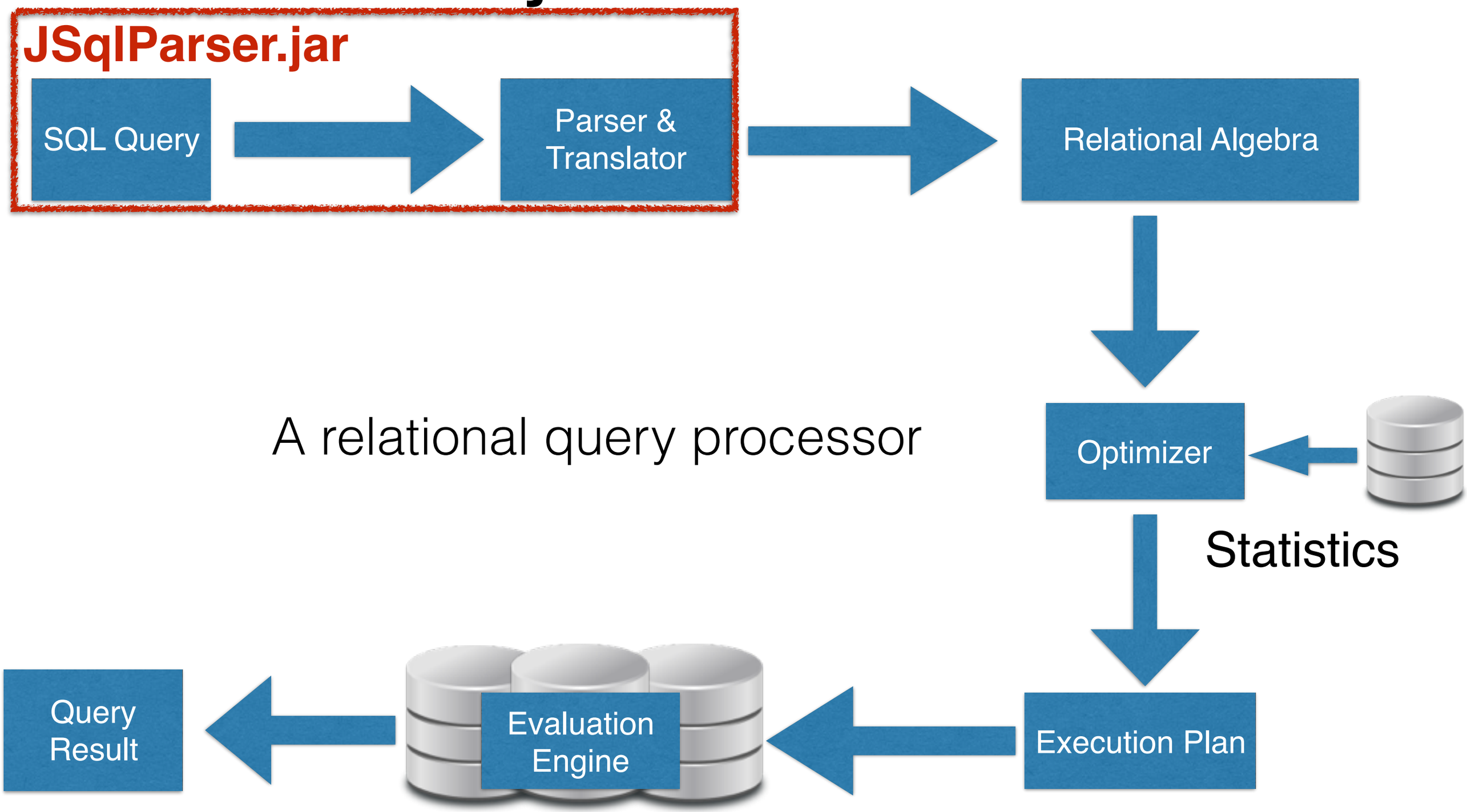
DμBStep emails your group

Project Outline



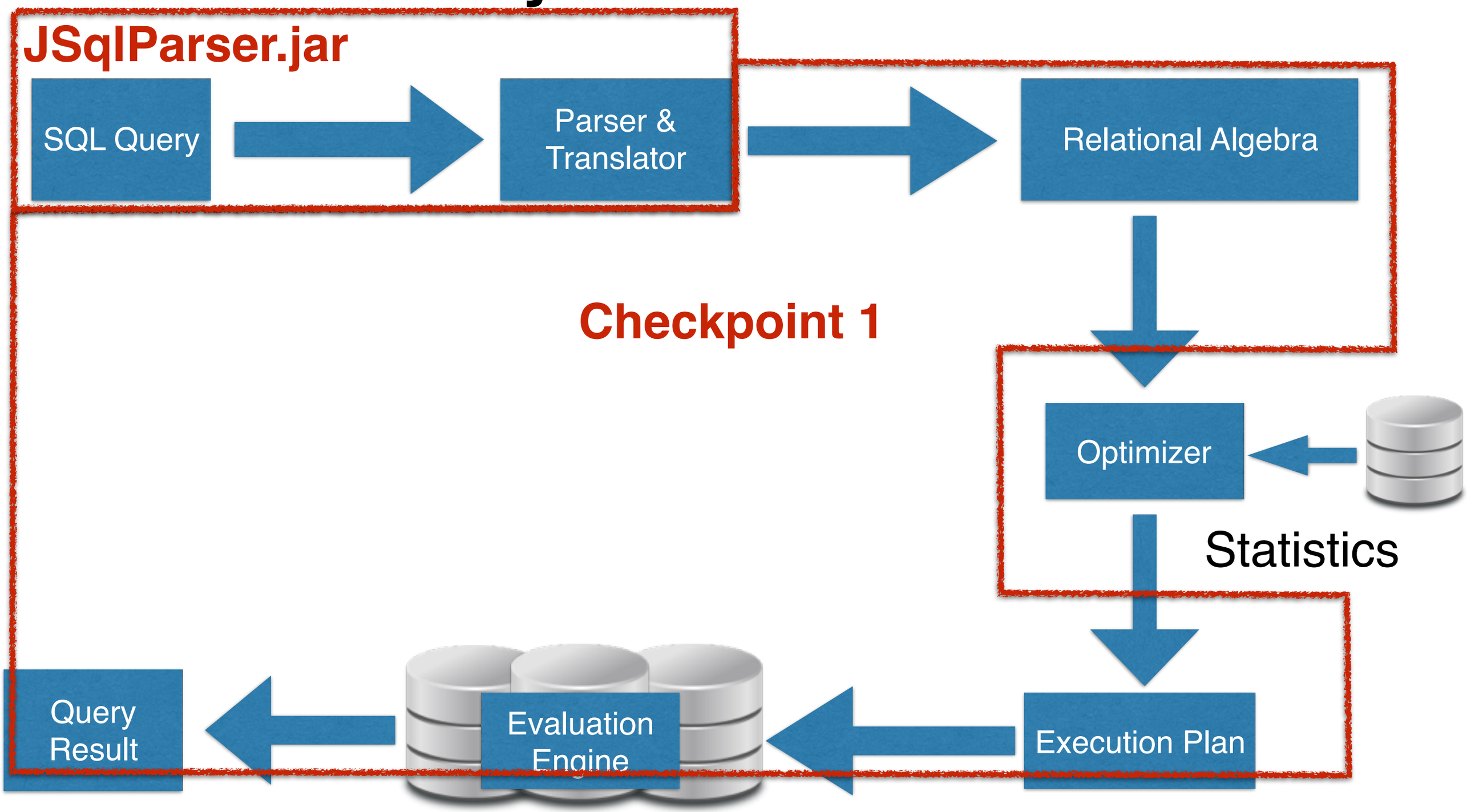
A relational query processor

Project Outline

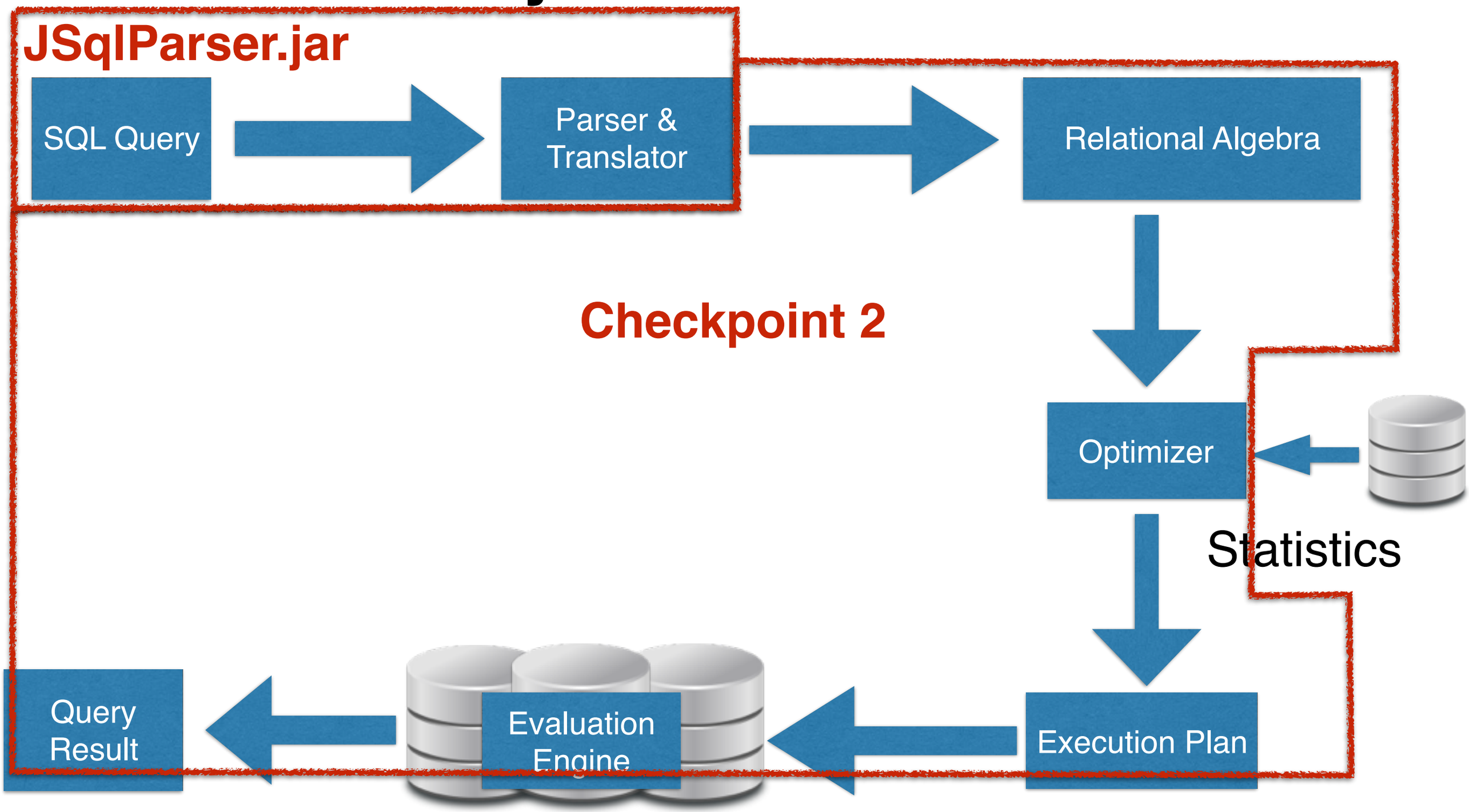


A relational query processor

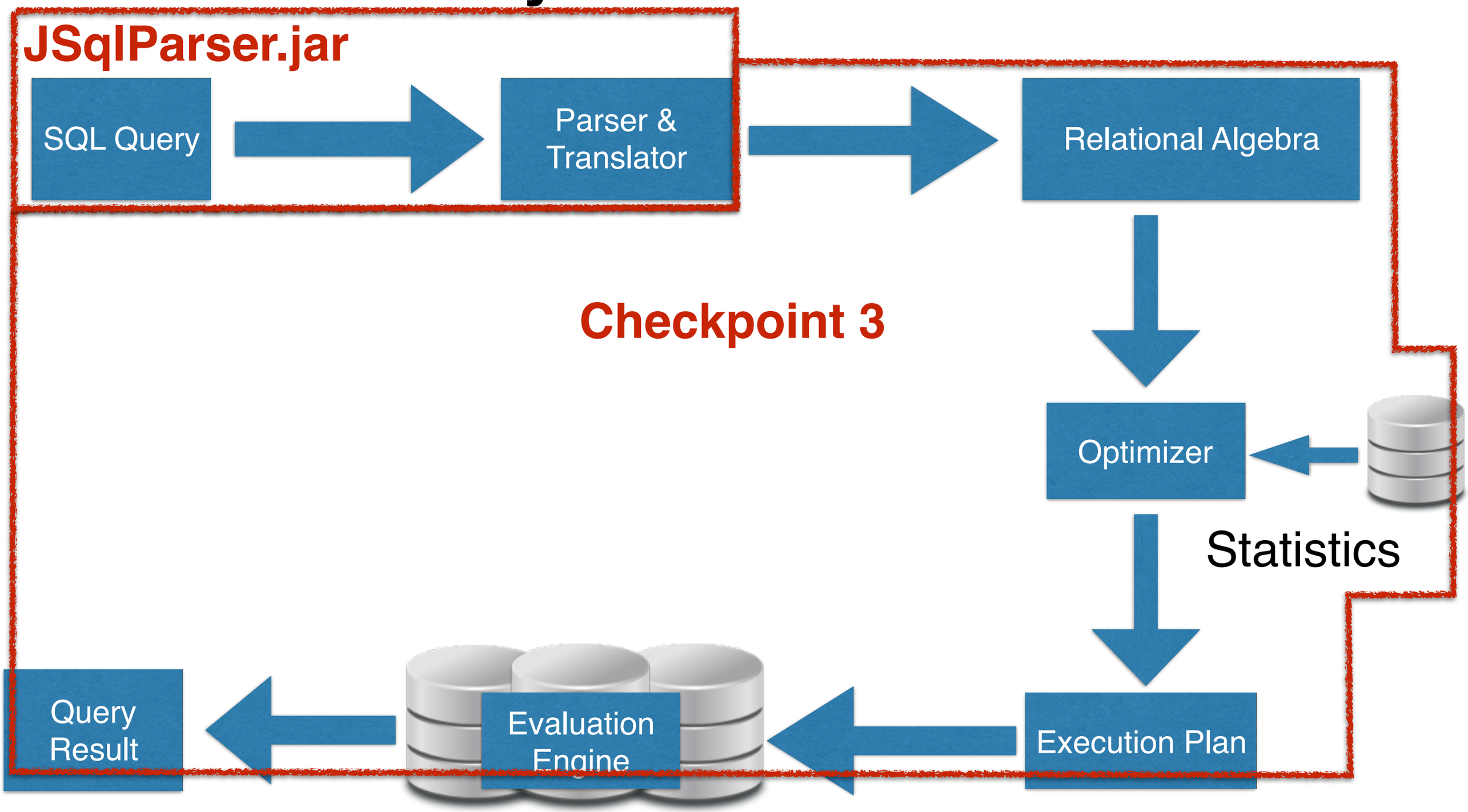
Project Outline



Project Outline



Project Outline



Projects

- **Checkpoint 0**: “Hello World” Set-up **(Due Feb 8)**
  - 5% of your overall grade (free points)
- **Checkpoint 1**: Basic SPJU Query Evaluation
  - 15% of your overall grade
- **Checkpoint 2**: “Big” Data & Query Optimization
  - 15% of your overall grade
- **Checkpoint 3**: Pre-computation
  - 15% of your overall grade

Those 5 free points sounded interesting…

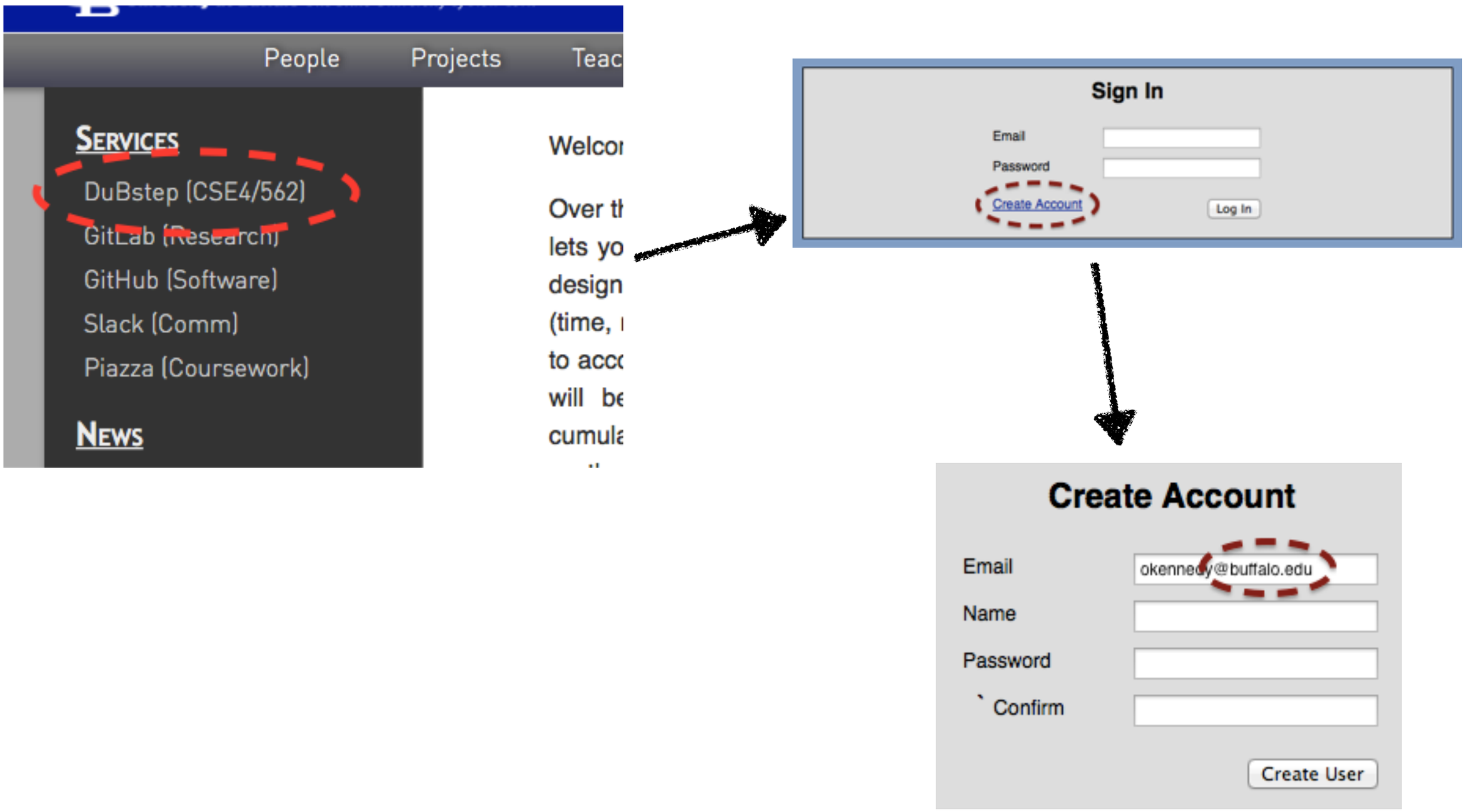
… what do I need to do to get them?

Those 5 free points sounded interesting…

… what do I need to do to get them?

http://odin.cse.buffalo.edu/dubstep/checkpoint0.html

5 free points



OMGWTFBBQTooHard

5 free points

- Create a group of up to 4 people.
- Register your group.
- Access your group's GIT repository.
- Commit a "Hello World" program.
- Hit "Submit"

If it doesn’t work, try again

Submit any project as many times as you need to
(before the deadline)

Your grade will not go down if you submit again

Any questions on the project?

Ways to Fail
(do not do these things)

- Start your project at the last minute
- Don’t go to office hours
- Don’t ask questions on Piazza
- Wait until the deadline to submit for the first time

Ways to Fail
(do not do these things)

- Start your project at the last minute
- Don’t go to office hours
- Don’t ask questions on Piazza
- Wait until the deadline to submit for the first time
- Cheat

Academic Integrity

Cheating is submitting any work that you did not perform by yourself as if you did.

References (be sure to cite properly):

Wikipedia, Wikibooks (or similar): **OK**

Public Code:

StackExchange (or similar): **NOT OK**

<u>Discussing</u> ideas with classmates out of class:

"A hash index has O(1) lookups": **OK** (except during exams 😇)

<u>Sharing</u> code or answers with classmates:

"Just have a look at how I implemented it": **NOT OK**

MOSS

Submission Overlap
(Ignoring Library Code)

Moss Results

Sat Mar 2 20:19:46 PST 2013

Options -l java -d -m 10

CSE 562 Project 1

[How to Read the Results | Tips | FAQ | Contact | Submission Scripts | Credits]

File 1		File 2		Lines Matched
[redacted]	(52%)	[redacted]	(52%)	2142
[redacted]	(84%)	[redacted]	(84%)	1536
[redacted]	(40%)	[redacted]	(40%)	1194
[redacted]	(29%)	[redacted]	(18%)	1163
[redacted]	(13%)	[redacted]	(19%)	822
[redacted]	(10%)	[redacted]	(9%)	569
[redacted]	(11%)	[redacted]	(10%)	660
[redacted]	(11%)	[redacted]	(16%)	616
[redacted]	(10%)	[redacted]	(7%)	513
[redacted]	(10%)	[redacted]	(8%)	613

MOSS-Details

Identical Code Structure

670-683		725-738	
3450-3480		3267-3298	
8443-8488		6488-6533	
7985-8003		5903-5921	

```
                break;
            }
            }
            break;
    }
    //Implementation for LEAF operators
    case LEAF: {
            switch (q.type) {
            //Implementation for NULLSOURCE operator
            case NULLSOURCE: {
                    Datum[] datum = new Datum[0];
                    query_res.data_list.add(datum);
                    break;
            }
            //Implementation for SCAN operator
            case SCAN: {

            ScanNode node = (ScanNode)q;
                    File file = tables.get(node.table).getFile();
            FileReader fir;
            BufferedReader reader;
            Datum[] datum;
            try {

                fir = new FileReader(file);
                reader = new BufferedReader(fir);
                String record;
                StringTokenizer st = null;
                while ((record = reader.readLine()) != null) {
                    datum = new Datum[node.schema.size()];
                    int i = 0;
                    st = new StringTokenizer(record, ",");
                    while (st.hasMoreElements()) {
                        String token = st.nextToken().trim();
                        switch (node.schema.get(i).type) {
                        case INT:
                            datum[i] = new Datum.Int(Integer.parseInt(t
                            break;
                        case FLOAT:
                            datum[i] = new Datum.Flt(Float.parseFloat(t
                            break;
                        case STRING:
                            datum[i] = new Datum.Str(token);
                            break;
                        case BOOL:
                            datum[i] = new Datum.Bool(new Boolean(toker
                            break;
                        default:
                            break;
```

```
import edu.buffalo.cse.sql.Schema.TableFromFile;
import edu.buffalo.cse.sql.data.Datum;
import edu.buffalo.cse.sql.plan.PlanNode;
import edu.buffalo.cse.sql.plan.ScanNode;

public class ScanMethod extends UnionMethod {

        public static void scanop(Map<String, Schema.TableFromFile> tables, PlanN
                        throws NumberFormatException, IOException {

                                ScanNode leafnode = (ScanNode) q;
                                File file =tables.get(leafnode.table).getFile();
                                FileReader fir;
                                BufferedReader reader;
                                Datum[] data;
                                int sizeofschema= leafnode.schema.size();

                fir = new FileReader(file);
                reader = new BufferedReader(fir);
        String record;
                        StringTokenizer st = null;
                                while ((record = reader.readLine()) != nu
                                        data = new Datum[leafnode.schema.
                                        int i = 0;
                                        st = new StringTokenizer(record,
                                        while (st.hasMoreElements()) {
                                                String token = st.nextTok
                                                switch (leafnode.schema.g
                                                case INT:
                                                        data[i] = new Dat
                                                        break;
                                                case FLOAT:
                                                        data[i] = new Dat
                                                        break;
                                                case STRING:
                                                        data[i] = new Dat
                                                        break;
                                                case BOOL:
                                                        data[i] = new Dat
                                                        break;
                                                default:
                                                        break;
                                                }
                                                i++;
                                        }
                                        raparse.schemaVar.add(data);

                                }
                                fir.close();
```

Code in Case Statement

Code in “Operator Class”

Academic Integrity

Zero Tolerance: If I catch you submitting someone else's code, **you will fail the class**.

Group Responsibility: If your teammate cheats on a group project, **the entire group will be penalized**.

Share Code, Share Blame: If someone else submits your code as their own, **you will be penalized as well**.

Questions/Concerns?

What does a data-management system do?

Data Management

Analysis: Answering user-provided questions about a dataset

What kind of tools can we give end-users?

- Declarative Languages
- Organizational Datastructures (e.g., Indexes)

Manipulation: Safely persisting and sharing data updates

What kind of tools can we give end-users?

- Consistency Primitives
- Data Validation Primitives

Data

VS

Data

vs

```
{
    "firstName": "John",
    "lastName": "Smith",
    "age": 25,
    "address": {
        "streetAddress": "21 2nd Street",
        "city": "New York",
        "state": "NY",
        "postalCode": 10021
    },
    "phoneNumbers": [
        {
            "type": "home",
            "number": "212 555-1234"
        },
        {
            "type": "fax",
            "number": "646 555-4567"
        }
    ]
}
```

Data

vs

Databases exploit the data's structure!

```
{
    "firstName": "John",
                 "Smith",

                 ddress": "21 2nd Street",
                 "New York",
                  "NY",
        "postalCode": 10021
    },
    "phoneNumbers": [
        {
            "type": "home",
            "number": "212 555-1234"
        },
        {
            "type": "fax",
            "number": "646 555-4567"
        }
    ]
}
```

So let’s talk structure…

Types

Types

Integer

Floating Point Number

String

List/Array

Bag

Set

Struct

Dictionary/Object

Types

Primitive

Integer

Floating Point Number

String

Collection

List/Array

Bag

Set

Tuple

Struct

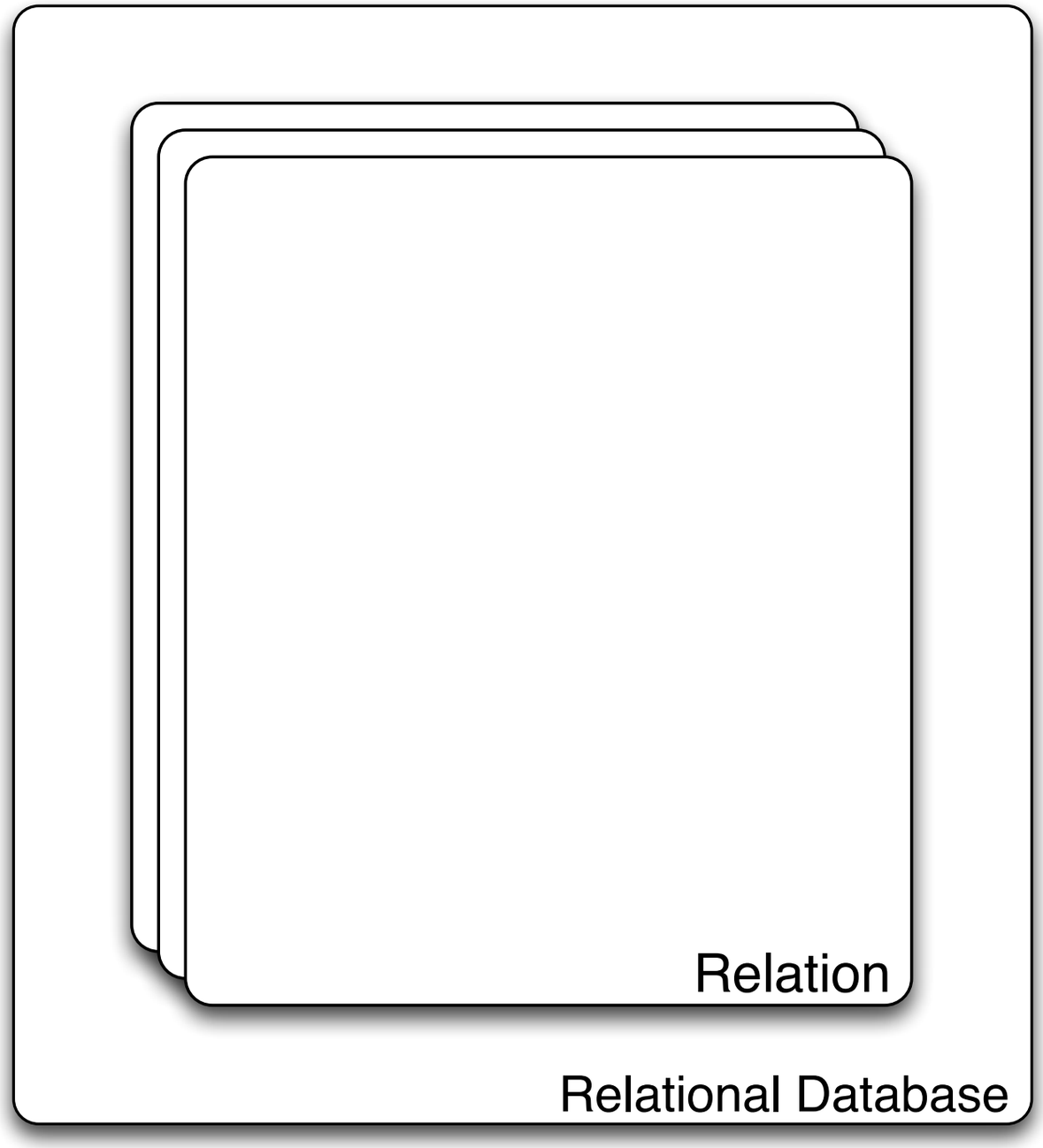
Dictionary/Object

Type Glossary

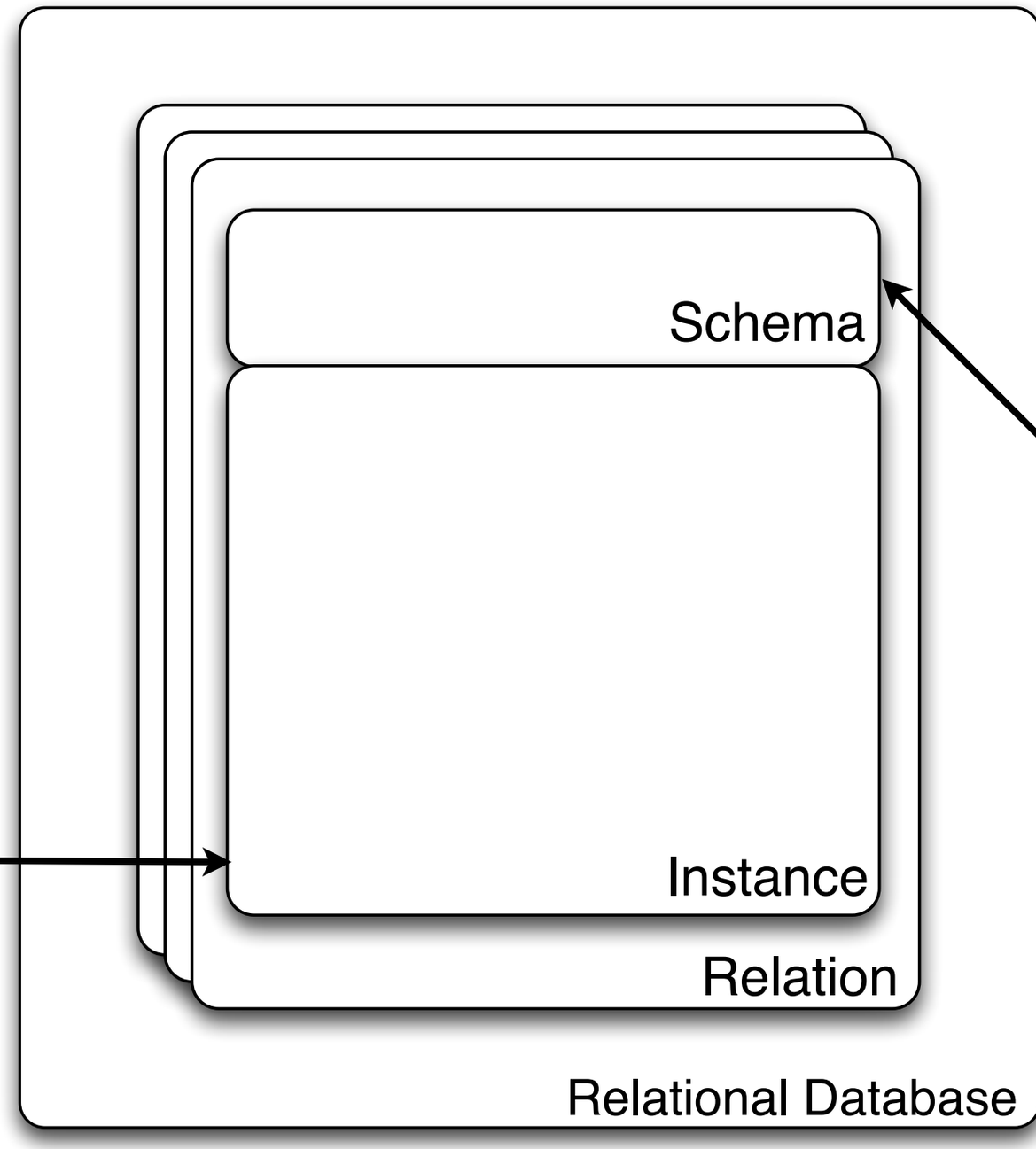
- **Primitive**: Basic building blocks like Int, Float, Char, *String*
- **Tuple**: Several ‘fields’ of different types. (N-Tuple = N fields)
  - A Tuple has a ‘schema’ defining names/types for each field
- **Set**: A collection of <u>unique</u> records, all of the same type
- **Bag**: An <u>unordered</u> collection of records, all of the same type
- **List**: An <u>ordered</u> collection of records, all of the same type

Relational Database Glossary

Relational Database Glossary



Relational Database Glossary



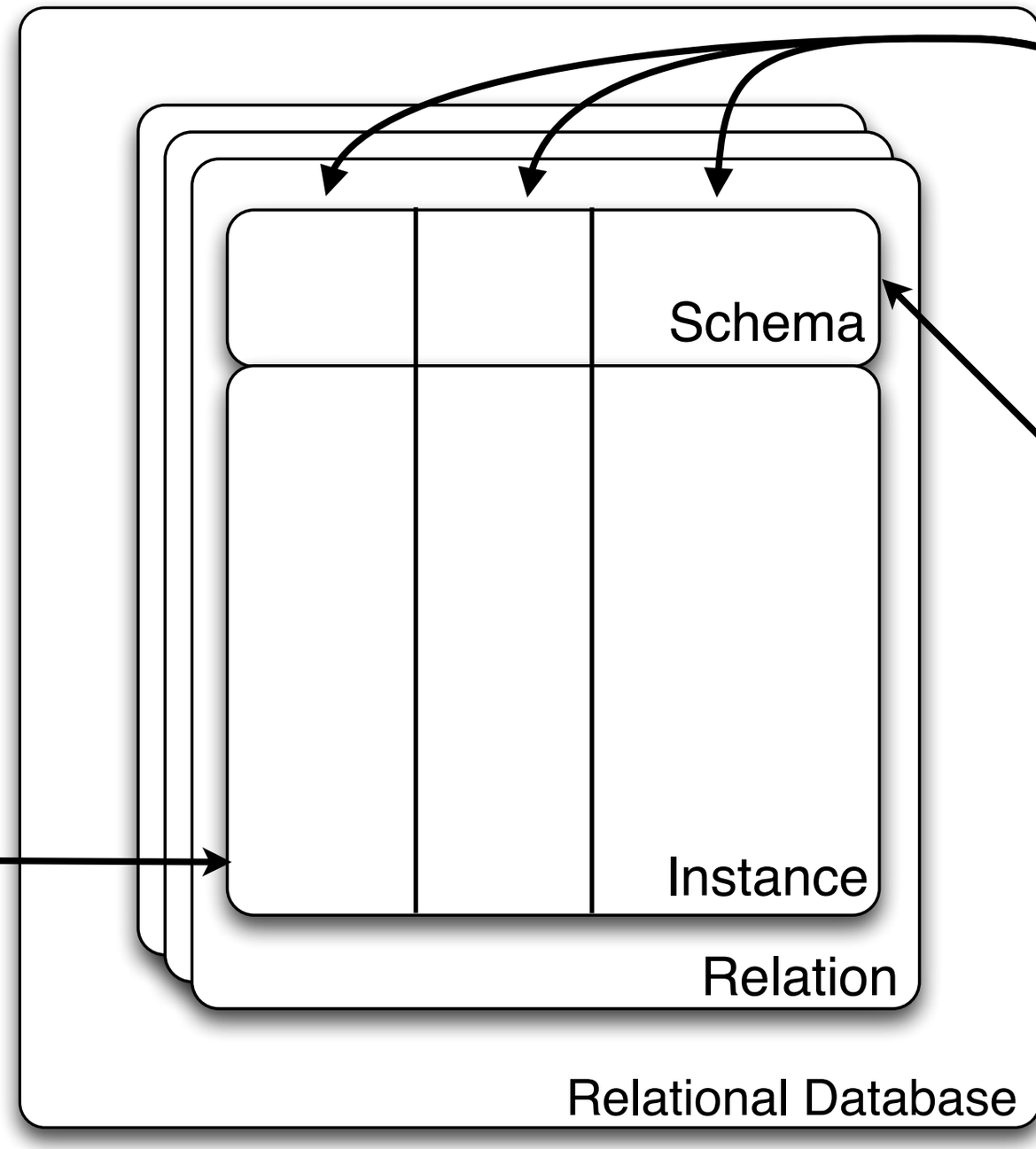
Specifies the name of the relation, plus the name and type of each column

```
Officers(
  firstname: string,
  lastname: string,
  id: int
)
```

The Data

```
[James,    Kirk,   2260]
[Jean Luc, Picard, 2360]
[Benjamin, Sisko,  2365]
```

Relational Database Glossary



Columns
(# = degree/arity)

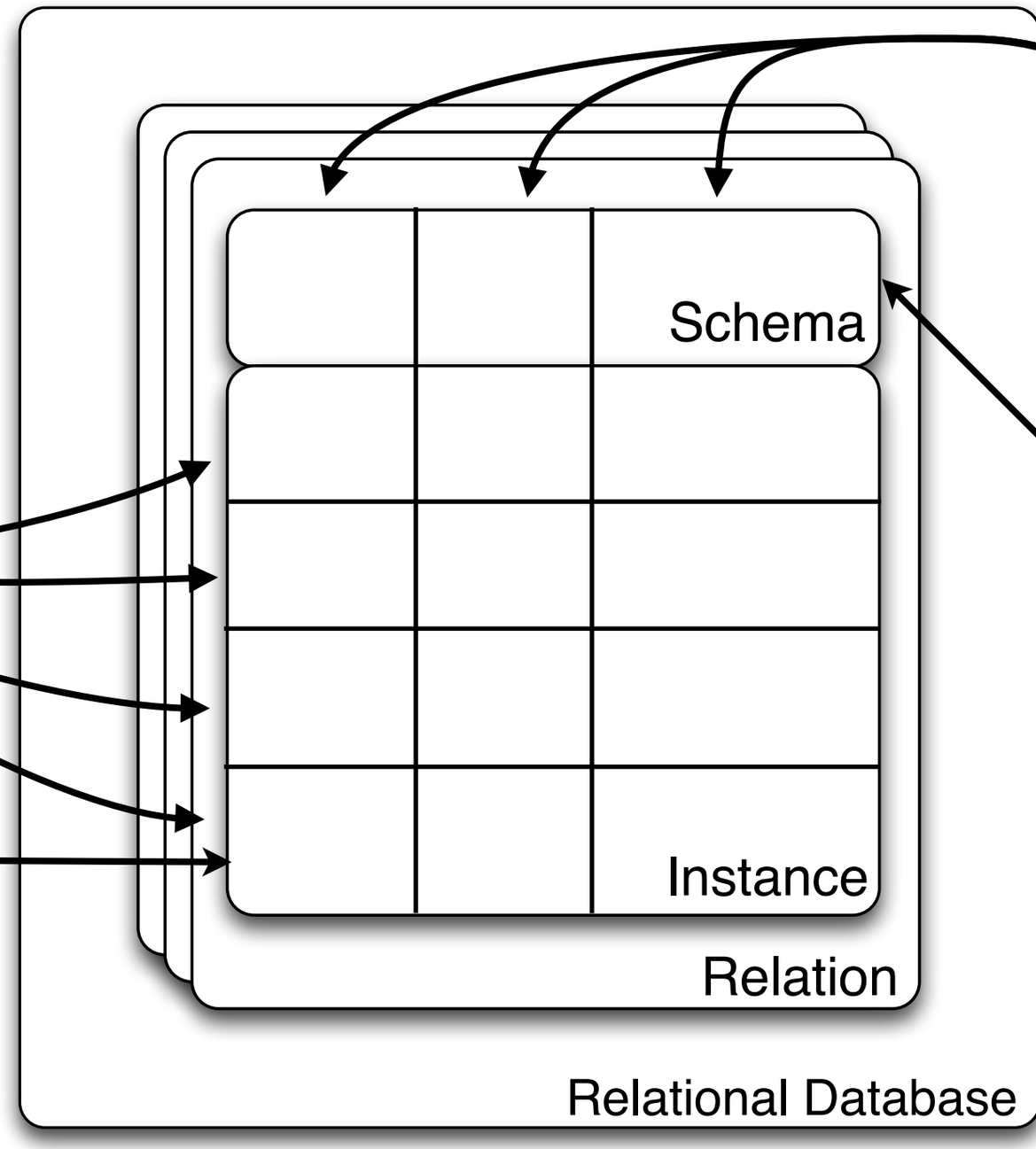
Specifies the name of the relation, plus the name and type of each column

The Data

```
[James,    Kirk,   2260]
[Jean Luc, Picard, 2360]
[Benjamin, Sisko,  2365]
```

```
Officers(
  firstname: string,
  lastname: string,
  id: int
)
```

Relational Database Glossary



Rows
(# = cardinality)

Columns
(# = degree/arity)

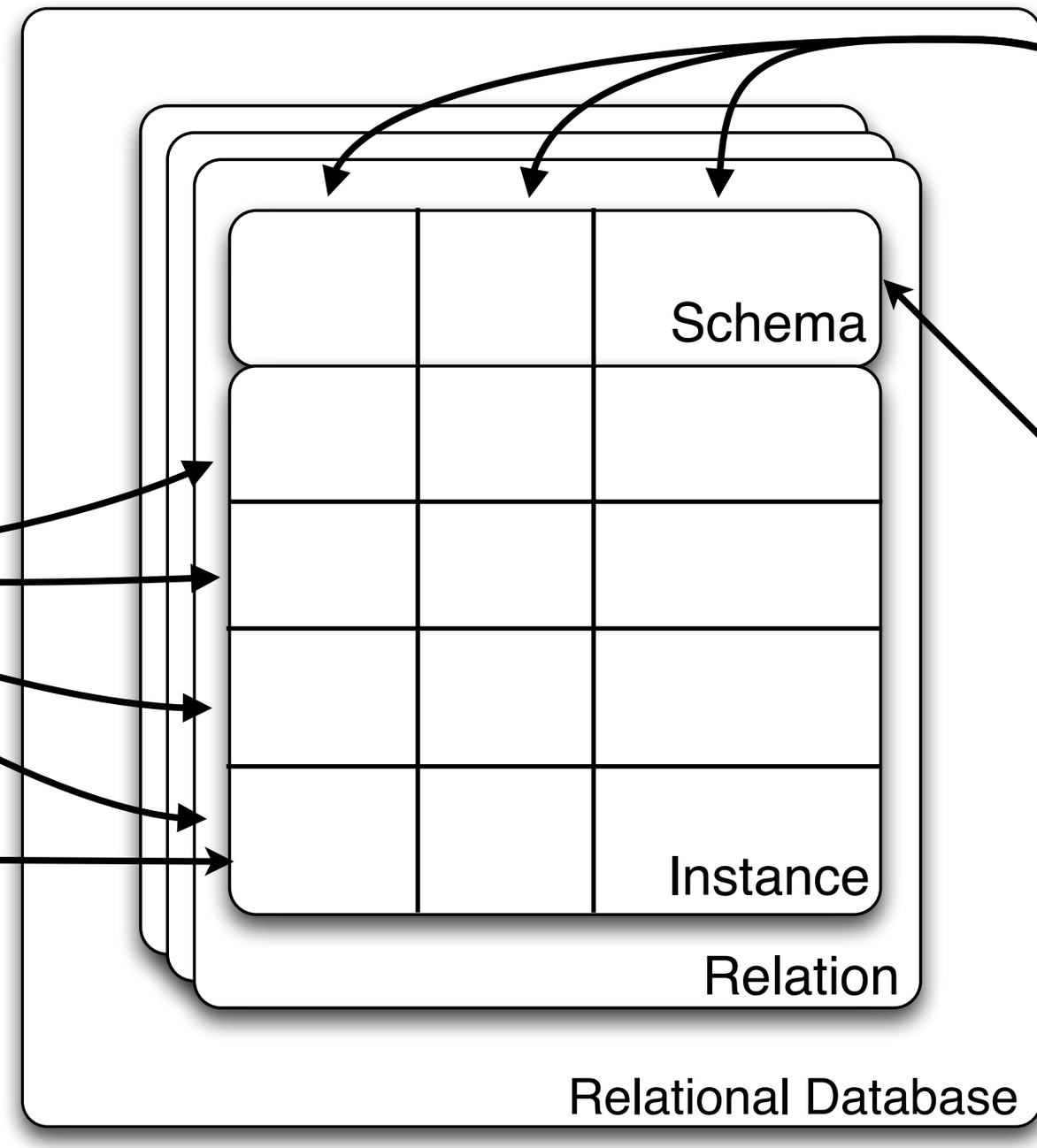
Specifies the name of the relation, plus the name and type of each column

The Data

```
[James,    Kirk,   2260]
[Jean Luc, Picard, 2360]
[Benjamin, Sisko,  2365]
```

```
Officers(
  firstname: string,
  lastname: string,
  id: int
)
```

Relational Database Glossary



Rows
(# = cardinality)

Columns
(# = degree/arity)

Specifies the name of the relation, plus the name and type of each column

The Data

```
[James,    Kirk,   2260]
[Jean Luc, Picard, 2360]
[Benjamin, Sisko,  2365]
```

```
Officers(
  firstname: string,
  lastname: string,
  id: int
)
```

A relation is a **set** of **tuples** (rows) with the same schema

Why?

Your data is currently an
Unordered Set of 100-attribute *Tuples*

Tomorrow, you'll be repeatedly asked for <u>1 specific attribute</u>
of <u>5 specific rows</u> identified by the <u>first attribute</u>

Can you do better?

Why?

Better Idea: Rewrite data into a 99-Tuple of Maps keyed on the 1st attribute

This representation is equivalent, and better for your needs.

Declarative specs make it easier to find equivalences.