Cosette: An Automated Solver for SQL

Shumo Chu, Chenglong Wang, Konstantin Weitz, Alvin Cheung, and Dan Suciu
Automated Solver for SQL: 
Q1 = Q2?
Motivation

• SQL is great
• A restricted abstraction enabling powerful optimizations
• Goal: formally reason SQL equivalences with automation:
  • Verify/find bugs in query optimization
  • Test generation
  • Auto grading
  • …
Challenges

• Deciding the equality of two relational queries are undecidable

• Rich language features
  • Aggregation and Group By
  • Index
  • Correlated Subqueries
  • Foreign keys
  • ……
Cosette: Coq + Rosette

• An (almost) automated solver for SQL by combining constraint solver and proof assistant

SQL: Q1 = Q2?

- Constraint Generator
  - Constraints
    - SMT Solver
    - Z3
    - Not Equal: Counter Examples

- UniNomial Generator
  - UniNomials
  - Proof Assistant
  - Equivalent
Finding Counter Examples with SMT Solver

SQL: $Q_1 = Q_2$?

Constraint Generator

Equal: Increase Model Size

SMT Solver

Constraints

Not Equal: Counter Examples
Encoding SQL

• A tuple as a list
  Tuple := List <Integer>

• A relation as a bag
  Relation := List <Pair<Tuple, Integer>>

• A SQL query as operations over symbolic values
Encoding SQL

```
SELECT pnum FROM Parts
WHERE qoh =
  (SELECT COUNT(shipdate)
   FROM Supply
   WHERE Supply.pnum = Parts.pnum
   AND shipdate < 10);
```

A SQL query

```
Parts = [([sv0,sv1],sv2), ([sv3,sv4],sv5)]
Supply = [([sv6,sv7],sv8)]

(assert r[0] =
  (if (sv1 = subQ1(([sv0,sv1],sv2))
    then ([sv0],sv2)
    else (if (sv4 = subQ1(([sv3,sv4],sv5))
      then ([sv3],sv5) else Nil))
  ... ...
```

SMT Constraints
Example: The Count Bug

• An infamous query optimization bug (Kim, W. ACM Trans. Database System 1982)

```
SELECT pnum FROM Parts
WHERE qoh =
  (SELECT COUNT(shipdate)
   FROM Supply
   WHERE Supply.pnum = Parts.pnum
   AND shipdate < 10);
```

Q1

```
WITH Temp AS
  SELECT pnum, COUNT(shipdate) AS ct
  FROM Supply
  WHERE shipdate < 10
  GROUP BY pnum
SELECT pnum FROM Parts , Temp
WHERE Parts.qoh = Temp.ct
  AND Parts.pnum = Temp.pnum;
```

Q2

Q1 and Q2 are not equal since Q2 ignores the cases when the count of a group is zero

Cosette:

<table>
<thead>
<tr>
<th>pnum</th>
<th>qoh</th>
<th>multiplicity</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>8</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>15</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>pnum</th>
<th>shipdate</th>
<th>multiplicity</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>0</td>
<td>2</td>
</tr>
</tbody>
</table>

Supply
What about equivalent queries?
Proving Equivalences with Proof Assistant

• Unbounded verification with proof assistant

• SQL where relations are modeled as lists requires finding invariants

• Inspired by K-Relation, We developed SQL semantics that eases reasoning equivalences:

![Diagram: SQL to UniNomial]
Proving Equivalences with Proof Assistant

SQL

UniNomial

\[ \begin{align*}
\text{a: Relation} & \quad [\text{a}]: \text{Tuple} \rightarrow \mathbb{N} \\
\text{b: Predicate} & \quad [\text{b}]: \text{Tuple} \rightarrow \{0, 1\} \\
\text{SELECT * FROM a WHERE b} & \quad \lambda t. \ [\text{a}] \ t \times [\text{b}] \ t \\
\text{a0 UNION ALL a1} & \quad \lambda t. \ [\text{a0}] \ t + [\text{a1}] \ t \\
\text{SELECT k FROM a} & \quad \lambda t. \ \sum_{t':\text{Tuple}} \text{if } t'.k = t.k \text{ then } [\text{a}] \ t \text{ else } 0
\end{align*} \]
Proving Equivalences with Proof Assistant

SQL | UniNomial

SQL equivalence | UniNomial equivalence

\[
\begin{align*}
\text{SELECT} & \quad \ast \\
\text{FROM} & \quad (a0 \ \text{UNION ALL} \ a1) \\
\text{WHERE} & \quad b
\end{align*}
\]

\[
(\text{SELECT} \ast \ \text{FROM} \ a0 \ \text{WHERE} \ b) \\
\quad \quad \text{UNION ALL} \\
(\text{SELECT} \ast \ \text{FROM} \ a1 \ \text{WHERE} \ b)
\]

Proof: function_extensionality; rewrite assoc_sum; reflexivity. Qed.
Evaluating Cosette

• **Bug**: 3 real-world optimizer bugs

• **XData**: query and mutant pairs collected from XData, a test generation framework

• **Exams**: a set of questions from the undergraduate data management class

• **Rules**: 23 query rewrite rules from database literatures and real-world optimizers

Unequal SQLs

Equivalent SQLs
## Evaluating Cosette

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Equiv?</th>
<th>Total Number</th>
<th>Automatically Decided</th>
<th>Interactively Decided</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>No.</td>
<td>Avg. Time</td>
<td>No.</td>
</tr>
<tr>
<td>Bugs</td>
<td>No</td>
<td>3</td>
<td>8.3 s</td>
<td>3</td>
</tr>
<tr>
<td>Exams</td>
<td>No</td>
<td>5</td>
<td>1.3 s</td>
<td>5</td>
</tr>
<tr>
<td>XData</td>
<td>No</td>
<td>9</td>
<td>&lt; 1 s</td>
<td>9</td>
</tr>
<tr>
<td>Rules</td>
<td>Yes</td>
<td>23</td>
<td>&lt; 1 s</td>
<td>17</td>
</tr>
<tr>
<td>Exams</td>
<td>Yes</td>
<td>4</td>
<td>&lt; 1 s</td>
<td>3</td>
</tr>
</tbody>
</table>

400 LOC to 15 LOC
Conclusions and Future Work

- Cosette: The first SQL solver combining SMT solver and proof assistant
- Automatically generating a verified query optimizer for new system
- Synthesize new optimization rules
- Website: cosette.cs.washington.edu
Why HoTT?

SQL

UniNomial

\[ a : \text{Relation} \quad \Rightarrow \quad [a] : \text{Tuple} \rightarrow \text{Type} \]

SELECT name FROM a

\[ \lambda t. \sum_{t':\text{Tuple}} \text{if } t'.name = t.name \text{ then } [a] t \text{ else } 0 \]

<table>
<thead>
<tr>
<th>name</th>
<th>salary</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;James&quot;</td>
<td>$10,000</td>
</tr>
<tr>
<td>&quot;Alex&quot;</td>
<td>$20,000</td>
</tr>
<tr>
<td>&quot;James&quot;</td>
<td>$30,000</td>
</tr>
<tr>
<td>&quot;Alex&quot;</td>
<td>$40,000</td>
</tr>
<tr>
<td>&quot;Alex&quot;</td>
<td>$50,000</td>
</tr>
</tbody>
</table>