Exercise

• Construct a knowledge base containing the following facts:
  • “Giants eat people.”
  • “Giants eat bunnies.”
  • “Bunnies eat grass.”
  • “People eat bunnies.”
  • “People eat people.”
  • “Those who are eaten by others hate those others.”
  • “Monsters love those who hate themselves.”
Then supply a query that can answer:
  • “Who do monsters love?”

Domain Specific Languages

• A domain specific language (DSL) is a language designed to solve a small set of tasks.
• DSLs frequently sacrifice expressiveness in favor of ease of use.
Examples

• SQL, used to manipulate tabular data
• HTML, used to represent web documents
• Verilog, a hardware description language
• PCRE, a string matching language based on regular expressions
• Make, used to describe software build dependencies
• XPath, used to query XML
• Postscript, used to describe printed documents
• LaTeX, used to cause undergraduates great anguish

Examples

• SQL cannot be used to perform arbitrary calculations; until recently (1999), it could not compute “reachability queries.”
• E.g., the transitive closure of “city y is reachable from city x via direct flight” computes the set of all cities reachable from city x.
• Why? Presumably the designers intended to disallow expensive queries.
• In fact, SQL is not Turing Complete. All SQL programs halt, by design.

Examples

• Postscript is Turing Complete.
• But writing arbitrary programs is a huge pain.
  • Commands are in Reverse Polish Notation (“operands first”)
  • The stack must be explicitly maintained
  • No user-defined types, etc.
• Great for offloading complex print jobs to a printer, though!
• Laser printers often ship with highly optimized Postscript interpreters (Raster Image Processor)

Completeness

• A formal system is a logical system for generating formulas.
• A formal system is complete with respect to a property if all formulas having that property can be derived using the rules (axioms) of the system.
Soundness

• A formal system is sound with respect to a property if all derivable formulas are true.

Incompleteness Theorem

• Kurt Gödel proved that mathematics (i.e., mathematical logic) cannot be both sound and complete wrt “provability.”
• Either:
  • you can define a formal system in which you can derive all the true mathematical statements, but which also admits false statements (inconsistent), or
  • you can define a formal system in which all statements are true, but in which you cannot derive all the true mathematical statements (incomplete).
• https://youtu.be/O4ndlDcDSGc

SQL

• SQL is a DSL for querying data, invented by E. F. Codd in 1970.
• SQL limits itself to only certain kinds of queries.
• All of those queries can be answered efficiently (and by implication, they terminate).
• The language is based on a theory of data and data queries called the relational algebra.
• The relational algebra lets users efficiently query data in a form that is largely independent of the organization of the data on disk.
• This was considered a major breakthrough when it was invented.
• For many practical reasons, SQL has diverged somewhat from the relational algebra.

Relational Algebra

• The relational algebra is based on set theory.
• A relation R is a set of tuples.
• Remember that sets contain only unique elements.
• Also, the order of elements in a set does not matter.
• The members of a tuple are called attributes.
• Note that the order of attributes in a tuple does not matter.
• We often think of relations as tables. But since relations are really sets of tuples, the order of attributes and rows in a table does not matter.
• A schema is the set of all defined relations.
• A database is a collection of instances of relations for a given schema.
Relational Algebra: Closure Property

- All operations in the relational algebra are closed, meaning that every operation on a relation yields a relation.
- Primitive operations:
  - Projection
  - Selection
  - Rename
  - Cartesian Product
- Complex operations:
  - Join, etc.

Relational Algebra: Projection

- Projection selects columns of a relation.
- Formally,
  \[ \Pi_{a_1, \ldots, a_n}(R) \]
  yields the relation \( R' \) such that \( R' \) is restricted on attributes \( a_1, \ldots, a_n \).
- For example, \( \Pi_{\text{Name}, \text{DeptName}}(\text{Employee}) \) yields

<table>
<thead>
<tr>
<th>Name</th>
<th>EmpId</th>
<th>DeptName</th>
</tr>
</thead>
<tbody>
<tr>
<td>Harry</td>
<td>3415</td>
<td>Finance</td>
</tr>
<tr>
<td>Sally</td>
<td>2241</td>
<td>Sales</td>
</tr>
<tr>
<td>George</td>
<td>3401</td>
<td>Finance</td>
</tr>
<tr>
<td>Harriet</td>
<td>2302</td>
<td>Sales</td>
</tr>
</tbody>
</table>

Relational Algebra: Selection

- Selection selects rows of a relation.
- Formally,
  \[ \sigma_\varphi(R) \]
  yields the relation \( R' \) such that \( R' \) is restricted on the predicate \( \varphi \).
- For example, \( \sigma_{\text{EmpId} > 3000}(\text{Employee}) \) yields

<table>
<thead>
<tr>
<th>Name</th>
<th>EmpId</th>
<th>DeptName</th>
</tr>
</thead>
<tbody>
<tr>
<td>Harry</td>
<td>3415</td>
<td>Finance</td>
</tr>
<tr>
<td>Sally</td>
<td>2241</td>
<td>Sales</td>
</tr>
<tr>
<td>George</td>
<td>3401</td>
<td>Finance</td>
</tr>
<tr>
<td>Harriet</td>
<td>2302</td>
<td>Sales</td>
</tr>
</tbody>
</table>

Relational Algebra: Rename

- Rename renames an attribute in a relation.
- Formally,
  \[ \rho_{a/b}(R) \]
  yields the relation \( R' \) such that attribute \( a \) is renamed to \( b \) in \( R' \).
- For example, \( \rho_{\text{EmpId}/\text{Id}}(\text{Employee}) \) yields

<table>
<thead>
<tr>
<th>Name</th>
<th>EmpId</th>
<th>DeptName</th>
</tr>
</thead>
<tbody>
<tr>
<td>Harry</td>
<td>3415</td>
<td>Finance</td>
</tr>
<tr>
<td>Sally</td>
<td>2241</td>
<td>Sales</td>
</tr>
<tr>
<td>George</td>
<td>3401</td>
<td>Finance</td>
</tr>
<tr>
<td>Harriet</td>
<td>2302</td>
<td>Sales</td>
</tr>
</tbody>
</table>
Relational Algebra: Cartesian Product

- Cartesian product pairs every tuple in relation R with a tuple in relation S.
- Formally, \( R \times S \) yields the relation \( T = \{ (r_1, \ldots, r_n, s_1, \ldots, s_n) | (r_1, \ldots, r_n) \in R, (s_1, \ldots, s_n) \in S \} \)
- Note that this is not quite the set theory Cartesian product because tuples are “flattened”.
- For example, \( \text{Employee} \times \text{Dept} \) yields

<table>
<thead>
<tr>
<th>Employee</th>
<th>Dept</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>EmpID</td>
</tr>
<tr>
<td>Harry</td>
<td>3415</td>
</tr>
<tr>
<td>Sally</td>
<td>2241</td>
</tr>
<tr>
<td>George</td>
<td>3401</td>
</tr>
<tr>
<td>Harriet</td>
<td>2302</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Employee×Dept</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
</tr>
<tr>
<td>-------</td>
</tr>
<tr>
<td>Harry</td>
</tr>
<tr>
<td>Sally</td>
</tr>
<tr>
<td>George</td>
</tr>
<tr>
<td>Harriet</td>
</tr>
</tbody>
</table>

Relational Algebra: Natural Join

- A number of useful operations called “joins” can be defined in terms of the primitive projection, selection, rename, and Cartesian product operations we just defined.
- Natural join pairs every tuple in relation R with a tuple in relation S where they agree on an attribute value.
- Formally, \( R \Join_i S \) yields the relation \( T = \{ (r_1, \ldots, r_n, s_1, \ldots, s_n) | (r_1, \ldots, r_n) \in R, (s_1, \ldots, s_n) \in S \} \) where \( r_i = s_i \) for a given \( i \).

<table>
<thead>
<tr>
<th>Employee</th>
<th>Dept</th>
<th>Employee×Dept</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>EmpID</td>
<td>DeptName</td>
</tr>
<tr>
<td>Harry</td>
<td>3415</td>
<td>Finance</td>
</tr>
<tr>
<td>Sally</td>
<td>2241</td>
<td>Sales</td>
</tr>
<tr>
<td>George</td>
<td>3401</td>
<td>Finance</td>
</tr>
<tr>
<td>Harriet</td>
<td>2302</td>
<td>Sales</td>
</tr>
</tbody>
</table>

Optimizations

- Because relational algebra abstrats queries from data representation, data can be organized on disk in whatever manner is the most efficient.
- Furthermore, the actual execution plan can be computed dynamically to take full advantage of hardware resources.
- For example, selection and projection both produce smaller relations than their inputs. Thus, if a query combines joins and selection/projection, a query planner can reorder operations so that selection/projection come first, reducing the number of tuples that must be computed in a join. This optimization is called predicate pushdown.
- Efficient database implementation is an active area of research (VLDB, SIGMOD, etc).

Importance of SQL

- SQL is one of the most important and successful languages ever invented.
- E.F., Codd won a Turing Award for his work on the relational algebra and relational database management systems.
- As of 2017, relational database systems alone were a $50 billion market.