The Relational Model and Relational Algebra

Nothing is so practical as a good theory
Kurt Lewin, 1945
The relational model

- Overcame shortcomings of earlier database models
- Has a strong theoretical base
- Codd was the major developer
Problems with other models

- Programmers worked at a low level of detail
- No commands for multiple record processing
- Little support for ad hoc querying by users
Objectives of relational model research

- **Data independence**
  - Logical and physical models are separate

- **Communicability**
  - A simple model understood by programmers and users

- **Set-processing**
  - Increase programmer productivity
Relational model concepts

- Data structures
- Integrity rules
- Operators
Data structures

Domain
- A set of values all of the same data type
- All the legal values of an attribute
- Defines what comparisons are legal
- Only attributes from the same domain should be compared

The domain concept is rarely implemented
Data structures

- Relations
  - A table of n columns and m rows
  - A relation’s *cardinality* is its number of rows
  - A relation’s *degrees* is its number of columns
  - A relational database is a collection of relations
    - No explicit linkages between tables

*Cardinality is easy to change but not degrees*
Structures

- **Primary key**
  - A unique identifier of a row in a relation
  - Can be composite

- **Candidate key**
  - An attribute that could be a primary key

- **Alternate key**
  - A candidate key that is not selected as the primary key

- **Foreign key**
  - An attribute of a relation that is the primary key of a relation
  - Can be composite
Integrity rules

Entity integrity
- *No component of the primary key of a relation can be null*
- Each row in a relation is uniquely identified

Referential integrity
- *A database must not contain any unmatched foreign key values*
- For every foreign key there is a corresponding primary key
Relational algebra has 8 operators

- Restrict
- Project
- Product
- Union
- Intersect
- Difference
- Join
- Divide
Restrict

Extracts rows from a single relation

<table>
<thead>
<tr>
<th>A</th>
<th>W</th>
<th>X</th>
<th>Y</th>
<th>Z</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Project

Extracts columns from a single relation

<table>
<thead>
<tr>
<th>A</th>
<th>W</th>
<th>X</th>
<th>Y</th>
<th>Z</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Product

Creates a new relation from all possible combinations of rows in two other relations.

A

<table>
<thead>
<tr>
<th>V</th>
<th>W</th>
</tr>
</thead>
<tbody>
<tr>
<td>v1</td>
<td>w1</td>
</tr>
<tr>
<td>v2</td>
<td>w2</td>
</tr>
<tr>
<td>v3</td>
<td>w3</td>
</tr>
</tbody>
</table>

B

<table>
<thead>
<tr>
<th>X</th>
<th>Y</th>
<th>Z</th>
</tr>
</thead>
<tbody>
<tr>
<td>x1</td>
<td>y1</td>
<td>z1</td>
</tr>
<tr>
<td>x2</td>
<td>y2</td>
<td>z2</td>
</tr>
</tbody>
</table>

A TIMES B

<table>
<thead>
<tr>
<th>V</th>
<th>W</th>
<th>X</th>
<th>Y</th>
<th>Z</th>
</tr>
</thead>
<tbody>
<tr>
<td>v1</td>
<td>w1</td>
<td>x1</td>
<td>y1</td>
<td>z1</td>
</tr>
<tr>
<td>v1</td>
<td>w1</td>
<td>x2</td>
<td>y2</td>
<td>z2</td>
</tr>
<tr>
<td>v2</td>
<td>w2</td>
<td>x1</td>
<td>y1</td>
<td>z1</td>
</tr>
<tr>
<td>v2</td>
<td>w2</td>
<td>x2</td>
<td>y2</td>
<td>z2</td>
</tr>
<tr>
<td>v3</td>
<td>w3</td>
<td>x1</td>
<td>y1</td>
<td>z1</td>
</tr>
<tr>
<td>v3</td>
<td>w3</td>
<td>x2</td>
<td>y2</td>
<td>z2</td>
</tr>
</tbody>
</table>
Union

Creates a new relation containing rows appearing in one or both relations

Duplicate rows are automatically eliminated

Relations must be union compatible
Intersect

- Creates a new relation containing rows appearing in both relations
- Relations must be union compatible
Difference

Creates a relation containing rows in the first relation but not in the second

Relations must be union compatible
Join

 Creates a new relation from all combinations of rows satisfying the join condition

 A join B where W = Z

\[
\begin{array}{|c|c|}
\hline
A &  \\
\hline
V & W  \\
\hline
v1 & wz1  \\
\hline
v2 & wz2  \\
\hline
v3 & wz3  \\
\hline
\end{array} \quad \begin{array}{|c|c|c|}
\hline
B &  \\
\hline
X & Y & Z  \\
\hline
x1 & y1 & wz1  \\
\hline
x2 & y2 & wz3  \\
\hline
\end{array}
\]

\[
\begin{array}{|c|c|c|c|}
\hline
A \text{ EQUIJOIN} B &  \\
\hline
V & W & X & Y & Z  \\
\hline
v1 & wz1 & x1 & y1 & wz1  \\
\hline
v3 & wz3 & x2 & y2 & wz3  \\
\hline
\end{array}
\]
Divide

Is there a value in the X column of A (e.g., x1) that has a value in the Y column of A for every value of y in the Y column of B?

<table>
<thead>
<tr>
<th>A</th>
<th>Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td></td>
</tr>
<tr>
<td>x1</td>
<td>y1</td>
</tr>
<tr>
<td>x1</td>
<td>y2</td>
</tr>
<tr>
<td>x1</td>
<td>y3</td>
</tr>
<tr>
<td>x2</td>
<td>y1</td>
</tr>
<tr>
<td>x2</td>
<td>y3</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>B</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>y1</td>
<td></td>
</tr>
<tr>
<td>y2</td>
<td></td>
</tr>
</tbody>
</table>

A DIVIDE B

<table>
<thead>
<tr>
<th>X</th>
</tr>
</thead>
<tbody>
<tr>
<td>x1</td>
</tr>
</tbody>
</table>
A primitive set of operators

- Only five operators are required
  - Restrict
  - Project
  - Product
  - Union
  - Difference
Relational algebra is a standard for judging a data retrieval language

<table>
<thead>
<tr>
<th>Relational algebra</th>
<th>SQL</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Restrict</strong></td>
<td>SELECT * FROM A WHERE condition</td>
</tr>
<tr>
<td><strong>Project</strong></td>
<td>SELECT X FROM A</td>
</tr>
<tr>
<td><strong>Product</strong></td>
<td>SELECT * FROM A, B</td>
</tr>
<tr>
<td><strong>Union</strong></td>
<td>SELECT * FROM A UNION SELECT * FROM B</td>
</tr>
<tr>
<td><strong>Difference</strong></td>
<td>SELECT * FROM A WHERE NOT EXISTS (SELECT * FROM B WHERE A.X = B.X AND A.Y = B.Y AND ...)</td>
</tr>
</tbody>
</table>

1. Essentially, where all columns of A are equal to all columns of B
A complete relational database

A fully relational database supports
- structures (domains and relations)
- integrity rules
- a manipulation language

Most commercial systems are not fully relational because they do not support domains and integrity rules
- Classified as relationally complete
Codd’s commandments

1. The information rule
   All data must appear to be stored as values in a table

2. The guaranteed access rule
   Every value in a database must be addressable by specifying its table name, column name, and the primary key of the row in which it is stored

3. Systematic treatment of null values
   There must be a distinct representation for unknown or inappropriate data

4. Active on-line catalog on the relational model
   There should be an on-line catalog that describes the relational model
Codd’s commandments

5. The comprehensive data sublanguage rule
   There must be a relational language that supports data definition, data manipulation, security and integrity constraints, and transaction processing operations

6. The view updating rule
   The DBMS must be able to update any view that is theoretically updateable

7. High-level insert, update, and delete
   The system must support set-at-a-time operations

8. Physical data independence
   Changes to storage representation or access methods will not affect application programs
Codd’s commandments

9. Logical data independence
   Information preserving changes to base tables will not affect application programs

10. Integrity independence
    Integrity constraints should be part of a database's definition rather than embedded within application programs
    It must be possible to change integrity constraints without affecting any existing application programs

11. Distribution independence
    Introduction of a distributed DBMS or redistributing existing distributed data should have no impact on existing applications

12. The nonsubversion rule
    It must not be possible to use a record-at-a-time interface to subvert security or integrity constraints
A relational DBMS must be able to manage databases entirely through its relational capacities.

A DBMS is either totally relational or it is not relational.
Key points

- The relational model is theoretically grounded and practically relevant
- Relational algebra is the foundation of SQL
- A relational DBMS should satisfy a range of requirements to be fully relational