The Relational Operators

	symbol	name	#tables	Preserves schema?	Assoc/ Comm?
⋆	$\boldsymbol{\sigma}$	select	unary	yes	yes
⋆	$\boldsymbol{\pi}$	project	unary	no	yes
	$\setminus$	difference	binary	yes	no
	$\cup$	union	binary	yes	yes
	$\times$	cartesian product	binary	no	yes
	"Redundant"				
	$\cap$	intersection	binary	yes	yes
⋆	$\bowtie$	join	binary	no	yes (?)
	Miscellaneous				
	$\div$	divide	binary	no	no

Joins

The Hallmark of the Relational Model

Types of joins:

- **natural join**: attributes from the two relations with the same names are used to join
- **semi-join**: a natural join that also *projects* over the attributes of the left-hand relation
- **equi-join**: attributes to join are specified explicitly
- **theta-join** (also, θ-join or condition-join): can use other comparators besides '=' between join conditions, such as '$\neq$'
- **Outer-joins**
  - **left**: preserves unmatched rows from the left-hand table
  - **right**: preserves unmatched rows from the right-hand table
  - **full**: preserves unmatched rows from both tables

Notes: Outer-joins are not associative!

Natural, semi-, equi-, and theta-joins are collectively called **inner-joins** sometimes, to contrast them with outer-joins.

Relational Database Classification

Classification	Description of Features
fully relational	Supports all eight relational operators. Also enforces both entity and referential integrity rules.
relationally complete	Supports all eight relational operators, but not the integrity rules.
minimally relational	Supports only **select**, **project**, and **join**.
tabular	Supports only **select**, **project**, and **join**, and requires that all access paths be defined by the user.

The 12 Commandments by Codd

Or Codd's Relational Rules (1985)

0. **Relational Database Management** (The General Rule)

   A relational database software package must use only its relational capabilities to manage the information stored in the database.

1. **Information Representation**

   All information stored in a relational database must be represented only by data element values that are stored in the tables that make up the database. Associations between data elements must not be logically represented in any other way, such as by pointers from one table to another.

2. **Logical Accessibility**

   Every data element value stored in a relational database must be accessible by stating the name of the table it is stored in, the name of the column under which it is stored, and the

3. **Representation of Null Values**

   The database software must have a consistent method for representing null values. For example, null values for numeric values must be distinct from zero or any other numeric value, and null character strings must be distinct from strings of blanks or any other character values.

4. **Catalog Facilities**

   The logical description of a relational database must be represented in the same manner as ordinary data so that facilities of the relational database software can be used to maintain database descriptions.

5. **Data Languages**

   A relational software product may support many different types of languages for describing data and accessing the database. However, there must be at least one language that uses ordinary character strings to support

   - the definition of data,
   - the definition of views,
   - the manipulation of data,
   - constraints on data integrity,

- and the boundaries of recovery units.

6. **View Updatability**

   Any view that can be defined using combinations of base tables that are, in theory, updatable must be capable of being updated by the database software.

7. **Insert, Update, and Delete**

   Any operand that describes the results of a single retrieval operation must also be capable of being applied to a single insert, update, or delete operation as well.

8. **Physical Data Independence**

   Changes that are made to physical storage representations or access methods must not require changes to be made to application programs.

9. **Logical Data Independence**

   Changes that are made to tables that do not modify any of the data already stored in the tables must not require changes to be made to application programs.

10. **Integrity Constraints**

    Constraints that apply to entity integrity and referential integrity must be specifiable by the data language implemented by the database software and not by statements coded into application programs.

11. **Database Distribution**

    The data language implemented by the database software must support the ability to distribute the database without requiring changes to be made to application programs. This facility must be provided in the data language whether or not the database software itself supports distributed databases.

12. **Non-subversion**

    If the database software supports facilities that allow an application program to operate on tables a row at a time, an application program using this type of database access must be prevented from bypassing **entity integrity** or **referential integrity constraints** that are defined by the database.

References

1. Codd, E. F., "Is your DBMS Really Relational?", *Computerworld*, Oct. 14, 1985.
2. Codd, E. F., "Does your DBMS Run by the Rules?", *Computerworld*, Oct. 21, 1985.