Databases - Sets & Relations

Relational terminology

Much of the power of relational databases comes from the fact that they can be described and analysed mathematically.

In particular, queries can be expressed with absolute precision using either

- The relational algebra
  A *procedural* language expressing how to extract information from the database.

- The relational calculus
  A *declarative* language expressing what information should be extracted from the database.

Sets

A set is simply a collection of objects, and is usually denoted by a capital letter such as A.

A set can be specified just by listing all of its members explicitly

\[ A = \{ \text{George, John, James Thomas} \} \]

\[ B = \{ \text{red, green, blue} \} \]

\[ C = \{ 2, 4, 6, 8, 10 \} \]

or sometimes by specifying a *pattern*

\[ D = \{ 2, 4, 6, 8, \ldots \} \]

Membership

The symbol ∈ is used to denote membership of the set i.e. whether an object belongs to the set or not.

Therefore

\[ \text{George} \in A \]

but

\[ \text{violet} \notin B \]

Similarly

\[ 102 \in D \]

but

\[ 999 \notin D \]
Defining Properties

A more precise way of defining the set $D$ is to use a *defining property* of the set.

$$D = \{x \mid x \text{ is a positive even integer}\}$$

This way of defining a set is reminiscent of the *WHERE* statement of SQL.

Sets can be defined in terms of other sets like this

$$E = \{x \mid x \in D \text{ and } x \text{ is a square}\}$$

This syntax is now almost identical to an SQL statement.

Cardinality

The *cardinality* of a set is its size; some sets are *finite* in that they have a specific number of elements while some are *infinite*.

The sets $A$, $B$ and $C$ are all finite, while $D$ is infinite.

There is a very special set $\emptyset$ that is called the *empty set* because it contains *no elements*.

Clearly most of the sets that we deal with in databases are *finite*.

Subsets and Supersets

A set $X$ is a *subset* of a set $Y$, denoted

$$X \subseteq Y$$

if every member of $X$ is also a member of $Y$.

There are two special cases:

$$\emptyset \subseteq Y$$

and

$$Y \subseteq Y$$

If we want to express the fact that $X$ is a *proper* subset of $Y$ then we say

$$X \subset Y.$$
Intersection

Given two sets $X$ and $Y$, the intersection of $X$ and $Y$ is the set containing all the objects that are members of $X$ and members of $Y$. In symbols, we have

$X \cap Y = \{x \mid x \in X \text{ and } x \in Y\}$.

Therefore we have

$A \cap B = \emptyset$

and

$C \cap D = C$.

Cartesian Product

For database purposes, the Cartesian product is one of the most important set operations.

Given two sets $X$ and $Y$, the Cartesian product $X \times Y$ is the set of ordered pairs with components from $X$ and $Y$ respectively:

$X \times Y = \{(x, y) \mid x \in X, y \in Y\}$

Notice that the elements of $X \times Y$ are pairs.

Cartesian Product Example I

If

$A = \{\text{George}, \text{John}, \text{James Thomas}\}$

and

$F = \{\text{Washington}, \text{Adams}, \text{Jefferson}, \text{Madison}\}$

then $A \times F$ contains all of

$\langle \text{George, Washington} \rangle \quad \langle \text{George, Adams} \rangle \quad \langle \text{George, Jefferson} \rangle \quad \langle \text{George, Madison} \rangle$

$\langle \text{John, Washington} \rangle \quad \langle \text{John, Adams} \rangle \quad \langle \text{John, Jefferson} \rangle \quad \langle \text{John, Madison} \rangle$

$\langle \text{James, Washington} \rangle \quad \langle \text{James, Adams} \rangle \quad \langle \text{James, Jefferson} \rangle \quad \langle \text{James, Madison} \rangle$

$\langle \text{Thomas, Washington} \rangle \quad \langle \text{Thomas, Adams} \rangle \quad \langle \text{Thomas, Jefferson} \rangle \quad \langle \text{Thomas, Madison} \rangle$

If $X$ and $Y$ are both finite, then $X \times Y$ has $|X| \times |Y|$ elements.

Cartesian Product Extended

If $X$, $Y$ and $Z$ are three sets then the Cartesian product

$X \times Y \times Z$

defined by

$X \times Y \times Z = \{(x, y, z) \mid x \in X, y \in Y, z \in Z\}$

has ordered triples as its elements.

Sometimes we shorten $X \times X$ to $X^2$ and $X \times X \times X$ to $X^3$ and so on.
### Extended Cartesian Product Example

Suppose that

\[ X = \{0, 1, \ldots, 255\} \]

Then \( X^3 \) is the set of all triples

\[ \{(x, y, z) \mid 0 \leq x, y, z \leq 255\} \]

This set could represent, for example, the set of all RGB-colours.

---

### Cartesian Products in a DB

Consider a variable like `first_name` which has type `VARCHAR(15)`. This variable can take on a finite number of legal values, and the entire collection of legal values is called the domain of the variable — suppose we call this \( D \).

Then the entire range of possible `(first_name, last_name)` combinations is the set \( D \times D \).

The collection of `(first_name, last_name)` combinations that is actually used in the database is therefore just a subset of \( D \times D \).

---

### Relations

Finally, we are in a position to define a (binary) relation:

**Definition**

A binary relation between two sets \( X \) and \( Y \) is a subset of \( X \times Y \).

Given a binary relation \( R \subseteq X \times Y \)

we say that \( x \) is related to \( y \) in the relation if \( (x, y) \in R \).

---

### Relation Example I

Consider the following subset of our earlier example product:

- (George, Washington)
- (George, Adams)
- (George, Jefferson)
- (George, Madison)
- (John, Washington)
- (John, Adams)
- (John, Jefferson)
- (John, Madison)
- (James, Washington)
- (James, Adams)
- (James, Jefferson)
- (James, Madison)
- (Thomas, Washington)
- (Thomas, Adams)
- (Thomas, Jefferson)
- (Thomas, Madison)

Under this relation, two names are related if they form the name of one of the first four American presidents.
Relation Example 2

Let $X$ be the set of all positive integers, and define a relation as follows:

$$ R = \{(x, y) \mid x \text{ is a divisor of } y\}. $$

It follows that

$$(2, 4) \in R \quad (5, 100) \in R \quad (3, 997) \notin R$$

Relation Example 3

Let $X$ be the set of all issued UWA student numbers, and let $Y$ be the set of all units at UWA.

Then we can define a relation $\text{EnrolledIn}$ such that

$$ (s, c) \in \text{EnrolledIn} $$

if and only if the student with student number $s$ is enrolled in the unit with code $c$.

Thus a typical element of this relation might be something like

$$(10423884, \text{CITS3240}).$$

Cardinality

A binary relation $R \subseteq X \times Y$ is called one-to-one if

- Each $x \in X$ appears in exactly one pair $(x, y) \in R$
- Each $y \in Y$ appears in exactly one pair $(x, y) \in R$

One-to-many

What happens if we alter the relation to include the fifth president, James Adams?

This relation is now called one-to-many because it is possible for one element of $x$ to be related to many elements of $y$. 
Relations
Many-to-many
Frequently a relation is *many-to-many* such as the “is a divisor of” relation discussed earlier.

Entity Sets
An *entity* is an object or event in which we are interested — an entity is described using a set of *attributes*.

For example, we might use the attributes `student-id` and `name` to describe a student. Then each individual student has a particular *value* for each attribute.

An *entity set* is a collection of similar objects in that every entity in the set has the same attributes — not the same *values*, but the same attributes.

For example, the set of all students is an *entity set* with two attributes.

Diagrammatic Representation
An entity set is described by using a specific type of diagram; a rectangle represents the entity set, while ovals represent the attributes.

Entity Sets and Relations
Each attribute of an entity set has a *domain*, which is the complete range of legal values for that attribute.

So if $D_1$ is the set of all legal student numbers, and $D_2$ the set of all legal names, then a specific entity set can be described just as a particular subset of $D_1 \times D_2$.

Therefore an entity set with two attributes with domains $D_1$ and $D_2$ is nothing more than a binary relation between $D_1$ and $D_2$. 
Entity Sets in an RDBMS

The whole point of all of this terminology is that in a relational database an entity set with two attributes is represented by a binary relation.

<table>
<thead>
<tr>
<th>Student-id</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>30781233</td>
<td>John Smith</td>
</tr>
<tr>
<td>30721438</td>
<td>Emily Tan</td>
</tr>
<tr>
<td>30611345</td>
<td>Mark James</td>
</tr>
<tr>
<td>30551001</td>
<td>Jennifer Lee</td>
</tr>
</tbody>
</table>

The relation schema is the names and types of the columns, while the relation instance is the collection of rows.

Arity

Of course, most entity sets have more than two attributes so cannot be modelled by a binary relation.

More generally, if \( D_1, D_2, \ldots, D_n \) are \( n \) sets then a relation with arity \( n \) is defined to be a subset

\[
R \subseteq D_1 \times D_2 \times \cdots \times D_n.
\]

A relation with arity 2 is binary, while a relation with arity 3 is called ternary.

Ternary Example

Let \( S \) be the set of all students, \( U \) the set of all units and \( L \) the set of all lecturers.

Then we could define a ternary relation

\[
\text{Teaches} \subseteq L \times U \times S
\]

where

\((l, u, s) \in \text{Teaches}\)

if \( l \) is the lecturer for unit \( u \) being taken by student \( s \).

The information contained in a ternary relation cannot be represented just by binary relations between \( S, U \) and \( L \).