An Introduction to Relational Algebra

James McMurray

PhD Student
Department of Empirical Inference

09/12/2014

MAX-PLANCK-GESELLSCHAFT

Outline of talk

MAX-PLANCK-GESELLSCHAFT

What is Relational Algebra?

- Algebra for modelling and querying data stored in relational databases.
- A relational database consists of a set of named relations, which contain named and typed attributes, and the data stored in tuples.
- A relation is a table.
- An attribute is a column.
- A tuple is a row.
- Relational Algebra queries are compositional, so any query returns a relation, which can in turn be queried.

MAX-PLANCK-GESELLSCHAFT

What is a Relational Database?

- In the dark ages before relational databases, data would have been stored together in one table:

storeName	movieTitle	priceAtStore	movieRating
'CheapBay'	'Edge Of Tomorrow'	10	4
'DVD World'	'Elysium'	20	9
'DVD World'	'Gattaca'	10	8
'Prime'	'Edge Of Tomorrow'	20	4
'Prime'	'Elysium'	15	9

- The problem is that it is possible to introduce internal inconsistencies.
- If we change the rating for 'Edge Of Tomorrow' at CheapBay, we also must remember to do the same at Prime, because it is a property of the movie not the store.

MAX-PLANCK-GESELLSCHAFT

What is a Relational Database?

- In a relational database the stores and movies are separated in to relations:

DVD

movie	rating
'Edge Of Tomorrow'	4
'Elysium'	9
'Gattaca'	8

Store

name	movie	price
'CheapBay'	'Edge Of Tomorrow'	10
'DVD World'	'Elysium'	20
'DVD World'	'Gattaca'	10
'Prime'	'Edge Of Tomorrow'	20
'Prime'	'Elysium'	15

What is a Relational Database?

- This prevents the development of inconsistencies.
- The relational algebra defines how to operate on the relations.
- This allows us to obtain all the information we wish from the relations, whilst maintaining consistency.

MAX-PLANCK-GESELLSCHAFT

Database schema

- The schema of the database is the names of the relation, names and types of attributes - the structure of the database.
- Usually, the schema is fixed in advance, and the data instances may change with time.
- All attributes are typed - i.e. Ints, strings, etc., the implementation depends on the specific database management system (DBMS).
- In addition to typed values, there exists the null value which is used for missing values.
- A key is an attribute which is unique for every tuple (usually an ID, or row ID), this is important for indexing the relations in practice.
- Note there is a debate over where relations (tables) should have singular or plural names. Ruby on Rails uses plurals, Django uses singular names.

Example database

DVD

movie	rating
'Edge Of Tomorrow'	4
'Elysium'	9
'Gattaca'	8

A relation DVD, with the attributes movie (String) and rating (Int).

Store

name	movie	price
'CheapBay'	'Edge Of Tomorrow'	10
'DVD World'	'Elysium'	20
'DVD World'	'Gattaca'	10
'Prime'	'Edge Of Tomorrow'	20
'Prime'	'Elysium'	15

A relation Store, with the attributes name (String), movie (String) and price (Int).

MAX-PLANCK-GESELLSCHAFT

Relational Algebra operators

- The simplest query is just the relation name, which returns the relation (table).
- Operators act on this to filter, slice and combine the data.
- The projection operator, $\Pi_{[attr]}$, returns the relation with only the chosen attributes, from the relation to which it is applied.
- The selection operator, $\sigma_{[cond]}$, returns a relation with only the tuples matching the conditions provided, from the relation to which it is applied.
- Note we use $\wedge$ for AND, and $\vee$ for OR.

Example of projection and selection

DVD

movie	rating
'Edge Of Tomorrow'	4
'Elysium'	9
'Gattaca'	8

$\Pi_{\text{rating}}\text{DVD} =$

rating
4
9
8

$\sigma_{\text{rating}>=5}\text{DVD} =$

movie	rating
'Elysium'	9
'Gattaca'	8

MAX-PLANCK-GESELLSCHAFT

Example of combination

DVD

Movie	Rating
'Edge Of Tomorrow'	4
'Elysium'	9
'Gattaca'	8

$$\Pi_{\text{movie}}\left(\sigma_{\text{rating}>=5}\text{DVD}\right) =$$

movie
'Elysium'
'Gattaca'

Relational Algebra operators (contd.)

- The tuple cross-product, $\times$, returns a relation with every combination of the tuples.
- The natural join, $\bowtie$, is the cross-product combined with the selection operator to match tuples according to shared attribute names.
- Note any relation natural joined with itself, returns the same relation itself.
- Note the natural join just syntactic sugar for combining the cross-product and a specific selection operator.
- The Theta join, $\bowtie_\theta$, is the cross-product combined with a general selection operator.
- This is also just syntactic sugar, and is the standard JOIN ON operator in DBMSs.

MAX-PLANCK-GESELLSCHAFT

Example database

DVD

movie	rating
'Edge Of Tomorrow'	4
'Elysium'	9
'Gattaca'	8

A relation DVD, with the attributes movie (String) and rating (Int)

Store

name	movie	price
'CheapBay'	'Edge Of Tomorrow'	10
'DVD World'	'Elysium'	20
'DVD World'	'Gattaca'	10
'Prime'	'Edge Of Tomorrow'	20
'Prime'	'Elysium'	15

A relation Store, with the attributes name (String), movie (String) and price (Int)

Example cross-product

- The cross-product returns every possible combination of the tuples.

$$(\Pi_{\text{name}}\text{Store}) \times (\Pi_{\text{Movie}}\text{DVD}) =$$

name	movie
'CheapBay'	'Edge Of Tomorrow'
'CheapBay'	'Elysium'
'CheapBay'	'Gattaca'
'DVD World'	'Edge Of Tomorrow'
'DVD World'	'Elysium'
'DVD World'	'Gattaca'
'Prime'	'Edge Of Tomorrow'
'Prime'	'Elysium'
'Prime'	'Gattaca'

Example join

- The natural join combines the tuples in the relations, where the joint attribute (movie) is equal.

Store $\bowtie$ DVD $=$

name	movie	price	rating
'CheapBay'	'Edge Of Tomorrow'	10	4
'DVD World'	'Elysium'	20	9
'DVD World'	'Gattaca'	10	8
'Prime'	'Edge Of Tomorrow'	20	4
'Prime'	'Elysium'	15	9

Relational Algebra operators (contd.)

- The union operator, $\cup$, concatenates/stacks the tuples.
- The difference operator, $-$, subtracts the second set of tuples from the first .
- i.e. $A - B$ returns tuples in A but not in B.
- The intersection operator, $\cap$, returns the tuples present in both sets.
- Note the intersection operator is just syntactic sugar, we can always write:

$$A \cap B = A - (A - B)$$

MAX-PLANCK-GESELLSCHAFT

Example difference

- We want to find all stores which **only** sell movies for less than 12 dollars.
- We can do this by finding all stores which sell any movie for 12 dollars or more, and then subtracting this from the set of all stores.

$$(\Pi_{\text{name}} Store) \; - \; (\Pi_{\text{name}}(\sigma_{\text{price}>=12} \text{Store})) \quad =$$

name
'CheapBay'

MAX-PLANCK-GESELLSCHAFT

Relational Algebra operators (contd.)

- The rename operator, $\rho_{\text{[newnames]}}$, returns the relation with the attributes assigned to the new names.
- This is **not** just syntactic sugar.
- Technically we need attribute names to be the same to carry out the natural join.
- More importantly, it is necessary for disambiguation in self-joins.

$\rho_{\text{newname}}\ \Pi_{\text{rating}}\text{DVD} =$

newname
4
9
8

MAX-PLANCK-GESELLSCHAFT

Self-join example

- We want to find every distinct pair of names of movies which have been rated - i.e. (‘Edge Of Tomorrow’, ‘Elysium’)
- In order to do this we have to use a self-join, and the rename operator is critical (also note the importance of the greater than condition):

$$\sigma_{\text{movie}>\text{movie2}}\left((\Pi_{\text{movie}}\text{DVD}) \times (\rho_{\text{movie2}}\ \Pi_{\text{movie}}\text{DVD})\right) =$$

movie	movie2
‘Edge Of Tomorrow’	‘Elysium’
‘Edge Of Tomorrow’	‘Gattaca’
‘Elysium’	‘Gattaca’

MAX-PLANCK-GESELLSCHAFT

More complicated example

- We want to find the maximum rating of all films sold by Prime.
- The trick we use is to find the set of ratings which are ever less than another rating (via a self-join), and then subtract this from the set of all ratings of films sold by Prime (denoted by PrimeRatings).

$$\text{PrimeRatings} := \Pi_{\text{rating}}\left((\sigma_{\text{name}=\text{'prime'}}\text{Store}) \bowtie \text{DVD}\right)$$

$$\text{Result} := \text{PrimeRatings} - \Pi_{\text{rating}}\ \sigma_{\text{rating}<\text{rating2}}(\ \text{PrimeRatings} \times \rho_{\text{rating2}}\text{PrimeRatings}\ )$$

$=$

rating
9

MAX-PLANCK-GESELLSCHAFT

Even harder example

- We want to find the names of all the stores which sell **ALL** of the films which have ratings higher than 7.
- Trick is to calculate all possible, relevant store-movie tuples, then calculate the "could have been" tuples which were not observed, then find which stores these were present for, and subtract this from all the stores.
- This is **Relational Division**.

$$\mathit{GoodMovie} := \Pi_{\text{movie}}\ \sigma_{\text{rating}>7}\text{DVD}$$

$$\begin{aligned}\text{Result} :=& \Pi_{\text{name}}\text{Store} - \Pi_{\text{name}}(((\Pi_{\text{name}}\text{Store}) \times \mathit{GoodMovie}) \\ & - \Pi_{\text{name,movie}}\text{Store}) \\ =& \end{aligned}$$

name
'DVD World'

MAX-PLANCK-GESELLSCHAFT

Conclusion

- Relational algebra defines an algebra for querying relational databases.
- Operators may seem simple but queries can become complicated.
- Many other common extensions have been omitted here, such as the semijoin, antijoin and relational division - these are all syntactic sugar.
- In practice we do not query relational databases with relational algebra directly, but with a query language based upon it.
- The most popular is SQL, the Structured Query Language.
- Though there is a Relational Algebra interpeter called RA for SQLite (used for these examples).

Questions?

MAX-PLANCK-GESELLSCHAFT