CIS4301 Notes

Ryan Roden-Corrent

January 31, 2014

1 Relational Algebra

See `http://cise.ufl.edu/class/cis4301sp14/slides/ra.ppt` for class slides on this. These notes are mostly a condensed version of the slides, the slides contain some nice table images to help you visualize the operations.

1.1 What is Relational Algebra?

Operators most common actions you execute on relations

Operands relations or variables that represent relations

1.2 Core Relational Algebra

Union,Intersection,Difference most common actions you execute on relations

Selection picking certain rows

Projection picking certain columns

Products/Joins compsitions of relations

Renaming of relations and attributes

1.2.1 Selection

$R1 := \sigma_c(R2)$
C is a condition that refers to attributes or R2
R1 is all tuples of R2 that satisfy C

Relation Sells:		
bar	beer	price
Joe's	Bud	2.50
Joe's	Miller	2.75
Sue's	Bud	2.50
Sue's	Miller	3.00

$JoeMenu := \sigma_{bar="Joe's"}(Sells)$:

bar	beer	price
Joe's	Bud	2.50
Joe's	Miller	2.75
Sue's	Bud	2.50
Sue's	Miller	3.00

1.2.2 Projection

$R1 := \pi_L(R2)$
L is a list of attributes from R2's schema
R1 contains only the attributes of R2 listed in L (in the order they are listed)
Set operation: removes duplicate tuples

1.2.3 Extended Projection

$R1 := \pi_L(R2)$
Like projection, but L can contain arbitrary expressions involving attributes. Example: $\pi_{A+B->C,A,A}(R)$ will create a new table where the first column, C, is the sum of A and B, and the next two columns A1 and A2 are copies of the original A.

1.2.4 Product

$R3 := R1XR2$

- pair each tuple t1 or R1 with each tuple t2 of R2
- Concatenation t_1t_2 is a tuple of R_3
- Schema of R3 is the attributes of R1 and then R2, in order
- Beware attribute A of same name in R1 and R2: use R1.A and R2.A

Example: $R3 := R1XR2$
Duplicate column B, so use R1.B and R2.B to differentiate.

1.2.5 Theta Join

$R3 := R1 \bowtie_C R2$ Equivalent to taking the product $R1XR2$ and applying σ_C to the result.
$R \bowtie_\theta S \equiv \sigma_\theta(RXS)$
C can be any boolean-values condition.

1.2.6 Natural Join

Remove duplicate columns, only return columns that naturally combine. $R3 := R1 \bowtie R2$

1.2.7 Renaming

Gives a new schema to a relation.
$R1 := \rho_{R1(A_1,...,A_n)}(R2)$ or $R1(A_1, ..., A_n) := R2$ (simplified notation)
R1 is a relation with the same tuples as R2 but the attribtues $A_1, ..., A_n$.

1.3 Building Complex Expressions

Combine operators with parentheses and precedence rules.
Three notations:

1. Sequences of assignment statements
2. Expressions with several operators
3. Expression trees

1.3.1 Operator Precedence

1. $[\sigma, \pi, \rho]$ (highest)
2. $[X, \bowtie]$
3. $\cap$
4. $\cup$

1.3.2 Expression Trees

Leaves are operands (variables or constant relations).
Interior nodes are operators applied to children.
Example:

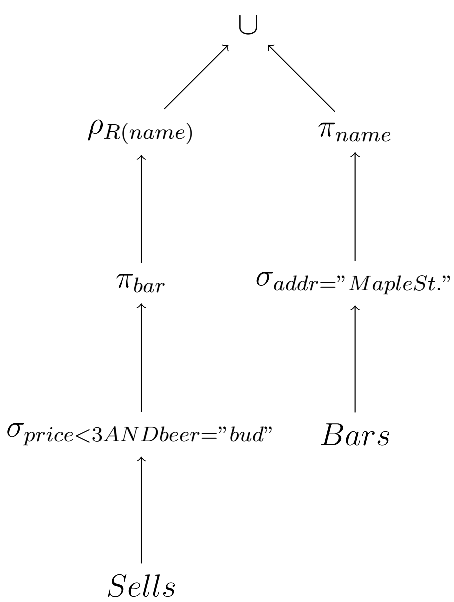

Figure 1: Find the names of all bars that are either on maple street or sell Bud for less than \$3

1.4 Relational Algebra on Bags

A bag is like a set, but duplicate elements are allowed. SQL is a bag language. Operations like projection are more efficient on bags than on sets.

1.4.1 Bag Union

Just add elements together, including duplicates.
$\{1, 2, 1\} \cup \{1, 1, 2, 3, 1\} = \{1, 1, 1, 1, 1, 2, 2, 3\}$

1.4.2 Bag Intersection

Minimum number of duplicates in resulting set.
$\{1, 2, 1, 1\} \cap \{1, 2, 1, 3\} = \{1, 1, 2\}$

1.4.3 Bag Difference

Result contains all tuples in first relation that aren't in the second.
$\{1, 2, 1, 1\} - \{1, 2, 3\} = 1, 1$

1.4.4 Bag Laws != Set Laws

Set union is **idempotent**, (result does not change if applied multiple times.)
For a bag union, if x appears n times in S, then it appears 2n times in union.

2 Why SQL?

Say what to do rather than how to do it.

SELECT desired attributes

FROM one or more tables

WHERE some condition holds

Beers desired attributes

FROM one or more tables

WHERE some condition holds

2.1 Example:

Names of beers made by Anheuser-Busch:

```
SELECT name
FROM Beers
WHERE manf = 'Anheuser-Busch'
```

name	manf
Bud	Anheuser-Busch

* indicates "all attributes in relation.

```
SELECT name
FROM Beers
WHERE manf = 'Anheuser-Busch'
```

Try adding things to SELECT *
Rename *name* to *beer* using **AS**

```
SELECT name AS beer, manf
FROM Beers
WHERE manf = 'Anheuser-Busch'
```

```
SELECT bar, ber,
price*114 AS PriceInYen
FROM Beers
```

Using Likes(drinker, beer): Put 'likes Bud' in attribute whoLikesBud

```
SELECT drinker
  'likes Bud' AS whoLikesBud
  FROM Likes
  WHERE beer = 'Bud'
```

2.2 Information Integration

Each **Data warehouse** manages several subdatabases that each have multiple tables.

Example: Each bar has its own relation Menu(beer,price) tat we need to integrate.

```
SELECT 'Joe''s Bar', beer, price
FROM Menu;
```

Do this for each bar to create a consistent schema, then take the Union.

2.3 Complex Conditions

Using Sells(bar, beer, price), find the price Joe charges for Bud.

```
SELECT price
FROM Sells
WHERE bar = 'Joe''s Bar' AND beer = 'Bud';
```

2.4 Patterns

Usable on **VARCHAR**. Useable on **DATE, INT**? Maybe.

```
<Attribute> LIKE <pattern>
<Attribute> iLIKE <pattern>   --case insensitive like
<Attribute> NOT LIKE <pattern>
```

% means any string

_ means any character

Example: choose numbers with two digits

```
FROM NUMBERS
WHERE number LIKE "__"
```

Example: choose numbers with at least two digits

```
FROM NUMBERS
WHERE number LIKE "__%"
```

Find drinkers with phone number starting with 555.

```
SELECT name
FROM Drinkers
WHERE phone LIKE '%555-_ _ _ _';
```

2.5 NULL Values

NULL could mean:

Missing Value we know it exists but dont have value

Inapplicable e.g. spouse of unmarried person

NULLs can be tricky, try to avoid if possible.

2.5.1 3-valued logic

TRUE = 1, FALSE = 0, UNKNOWN = 1/2
AND = MIN, OR=MAX, NOT(x) = 1-x
Ex:

```
TRUE AND (FALSE OR NOT(UNKNOWN)) = MIN(1, MAX(0, (1-1/2)))
```

Start on innermost paren and change truth values to numeric values. E.g. NOT(UNKNOWN) = (1-1/2)

2.5.2 Surprising Example

bar	beer	price
Joes	Bud	NULL

```
SELECT bar
FROM Sells
WHERE price < 2.00 OR price >= 2.00;
```

The comparisons become UNKNOWN because they are undefined for NULL.
Result is

```
NULL OR NULL
Max(1/2,1/2) = 1/2
```

Expected to get everything, got nothing

2.6 Multirelation Queries

Distinguish between attributes of same name by "relation.attribute"

2.6.1 Examples

Using Likes(drinker,beer), Frequents(drinker, bar), find the beers liked by at least one person who frequents Joe's.

```sql
SELECT beer
FROM Likes, Frequents
WHERE Bar = 'Joes' AND
  Frequents.drinker =
    Likes.drinker;
```

A theta join $\bowtie_=$ is used to equate the drinker attribute from each table. Can be done in any order.
For example: