UNIVERSITY OF ILLINOIS
AT URBANA-CHAMPAIGN

CS411 - Extended Relational Algebra and SQL: Queries

Announcements

- HW1 is due this Friday (submitted to Compass)
- MP1 will be released soon

Review

- How are queries represented internally?
- What kinds of constraints did we learn to express in relational algebra?

Correction

- Key constraints
  1. Rename two copies of the table
  2. Take the cross product of the relation with itself
  3. Select on the attributes for the key being equal ***and the others not being equal***
  4. Ensure that the result is the empty

Big picture

- Why are we studying relational algebra
  - essentially an abstract, formal DML
- Some practical aspects are not modeled by conventional relational algebra

Extending Relational Algebra

- We need to extend both the structure (operands) and the operators
  - structure: extend tuples from sets to *bags*
  - operators: add grouping, aggregation, and other new operators

Bags

- Also called “multisets”
- Generalize the concept of sets
- Members can appear more than once
  - relax uniqueness constraint of sets

Example

Person

First Name	Last Name	Phone	Email
Holden	Caufield	(217)-555-3251	nophoney@hotmail.com
Richard	Parker	(217)-555-1212	pi_delicious@gmail.com
Luke	Skywalker	(217)-555-2917	wompratbullseye@gmail.com
Marty	McFly	(217)-555-1987	delorian88@gmail.com
Richard	Parker	(217)-555-1212	pi_delicious@gmail.com
Luke	Skywalker	(217)-555-2917	wompratbullseye@gmail.com
Marty	McFly	(217)-555-1987	delorian88@gmail.com
Richard	Parker	(217)-555-1212	pi_delicious@gmail.com

Bags

- More efficient
  - Union or projection can require duplicate elimination
- Make new operations possible
  - Example: Average salary of people

$$AVERAGE(\pi_{Salary}(People))$$

  - This won't be correct if projection eliminates the duplicates

Example

$\pi_{Salary}(People)$

First Name	Last Name	Salary
Holden	Caufield	57,000
Richard	Parker	80,000
Luke	Skywalker	100,000
Marty	McFly	80,000

Average=79,250

Salary
57,000
80,000

verag 9,00

Set Operations

- Tuple t occurs m and times in R and and n times in S
- Union: $R \cup S$
  - – Each tuple t appears $n+m$ times
- Intersection: $R \cap S$
  - – Each tuple t appears min(n,m) times
- Difference: $R- S$
  - – Each tuple t appears max(0,m-n) times

Examples

Person1

First Name	Last Name
Holden	Caufield
Richard	Parker
Holden	Caufield

Person2

Last Name	First Name
Swan	Bella
McFly	Marty
Parker	Richard
Caufield	Holden

$Person1 \cup Person2$

First Name	Last Name
Holden	Caufield
Richard	Parker
Holden	Caufield
Bella	Swan
Marty	McFly
Richard	Parker
Holden	Caufield

Examples

Person1

First Name	Last Name
Holden	Caufield
Richard	Parker
Holden	Caufield

Person2

Last Name	First Name
Swan	Bella
McFly	Marty
Parker	Richard
Caufield	Holden

$Person1 \cap Person2$

First Name	Last Name
Holden	Caufield
Richard	Parker
Holden	Caufield

Examples

Person1

First Name	Last Name
Holden	Caufield
Richard	Parker
Holden	Caufield

Person2

Last Name	First Name
Swan	Bella
McFly	Marty
Parker	Richard
Caufield	Holden

$Person2 - Person1$

First Name	Last Name
Bella	Swan
Marty	McFly

Other operators

- Selection, Projection, Product, and Joins all work the same, but duplicates are not removed

Example

Album

AlbumTitle	BandName	DateReleased
Nevermind	Nirvana	09/24/1991
Nevermind	Nirvana	09/24/1991

Song

SongTitle	AlbumTitle	Length
Breed	Nevermind	3:03
Feel The Pain	Without a Sound	4:18
Feel The Pain	Without a Sound	4:18

$$Album \times Song$$

Album.AlbumTitle	BandName	DateReleased	SongTitle	Song.AlbumTitle	Length
Nevermind	Nirvana	09/24/1991	Breed	Nevermind	3:03
Nevermind	Nirvana	09/24/1991	Feel The Pain	Without a Sound	4:18
Nevermind	Nirvana	09/24/1991	Feel The Pain	Without a Sound	4:18
Nevermind	Nirvana	09/24/1991	Breed	Nevermind	3:03
Nevermind	Nirvana	09/24/1991	Feel The Pain	Without a Sound	4:18
Nevermind	Nirvana	09/24/1991	Feel The Pain	Without a Sound	4:18

Example

Album

AlbumTitle	BandName	DateReleased
Nevermind	Nirvana	09/24/1991
Nevermind	Nirvana	09/24/1991

Song

SongTitle	AlbumTitle	Length
Breed	Nevermind	3:03
Feel The Pain	Without a Sound	4:18
Lithium	Nevermind	4:17
Siva	Gish	4:21

$Album \bowtie Song$

AlbumTitle	BandName	DateReleased	SongTitle	Length
Nevermind	Nirvana	09/24/1991	Breed	3:03
Nevermind	Nirvana	09/24/1991	Lithium	4:17
Nevermind	Nirvana	09/24/1991	Breed	3:03
Nevermind	Nirvana	09/24/1991	Lithium	4:17

Extended operations

- δ - duplicate elimination
- Aggregation
  - SUM, AVG, MIN, MAX, COUNT
- γ - grouping
- π - extended projection
- τ - sorting
- $\overset{\circ}{\bowtie}$ - outerjoin

Duplicate Elimination

- $\delta(R)$
- Converts a bag into a set

Example

Person

First Name	Last Name	Salary
Holden	Caufield	50,000
Richard	Parker	60,000
Luke	Skywalker	70,000
Marty	McFly	40,000
Richard	Parker	60,000
Luke	Skywalker	70,000
Marty	McFly	40,000
Richard	Parker	60,000

$\delta(Person)$

First Name	Last Name	Salary
Holden	Caufield	50,000
Richard	Parker	60,000
Luke	Skywalker	70,000
Marty	McFly	40,000

Aggregation

- Summarize values of one attribute
- Applied to an attribute of a relation
  - e.g. SUM(SALARY)
- Most of them are obvious
  - e.g. MAX finds the maximum value
- COUNT is a bit different
  - Counts the number of values

Example

Person

First Name	Last Name	Salary
Holden	Caufield	50,000
Richard	Parker	60,000
Luke	Skywalker	70,000
Marty	McFly	40,000
Richard	Parker	60,000
Luke	Skywalker	70,000
Marty	McFly	40,000
Richard	Parker	60,000

SUM(SALARY)=450,000

AVG(SALARY)=56,250

MAX(SALARY)=80,000

MIN(SALARY)=40,000

COUNT(FirstName)=8

Grouping

- $\gamma_L(R)$
- L here is a list of
  - grouping attributes: attributes we want to gather tuples together
  - aggregation attributes: aggregation operators we apply to specific attributes
- Aggregation attributes are renamed with an arrow

Example

Person

First Name	Last Name	Genre	Salary
Holden	Caufield	Book	50,000
Richard	Parker	Book	60,000
Richard	Parker	Movie	23,000
Luke	Skywalker	Movie	70,000
Marty	McFly	Movie	40,000

Genre	minSalary
Book	50,000
Movie	23,000

$$\gamma_{genre, MIN(Salary) \rightarrow minSalary}(Person)$$

Extended Projection

- $\pi_L(R)$
- L consists of a list of:
  - Attributes from R
  - Expressions of the form x → y, which renames attribute x with name y
  - Expressions of the form E → y

Extended Projection

- Expressions of the form $E \rightarrow y$
  - E itself is a collection of expressions involving the attributes of the relation
    - addition
    - subtraction
    - string concatenation (written as ||)

Example

Person

First Name	Last Name	Genre	Salary	Age
Holden	Caufield	Book	50,000	16
Richard	Parker	Book	60,000	5
Richard	Parker	Movie	23,000	5
Luke	Skywalker	Movie	70,000	23
Marty	McFly	Movie	40,000	19

name	number
HoldenCaufield	50,016
RichardParker	60,005
RichardParker	23,005
LukeSkywalker	70,023
MartyMcFly	40,019

$$\pi_{firstName||lastName \rightarrow name, salary+age \rightarrow number}(Person)$$

Sorting

- $\tau_L(R)$
- Sorts the tuples of the relation
  - Rather than a bag of tuples, we now have a well ordered multiset of tuples
- L consists of a list of attributes
  - Sorted by the first attribute, ties are resolved by the second, further ties by the third, etc.

Example

Person

First Name	Last Name	Genre
Holden	Caufield	Book
Richard	Parker	Book
Richard	Parker	Movie
Luke	Skywalker	Movie
Marty	McFly	Movie

First Name	Last Name	Genre
Holden	Caufield	Book
Marty	McFly	Movie
Richard	Parker	Book
Richard	Parker	Movie
Luke	Skywalker	Movie

$$\tau_{LastName,FirstName,Genre}(Person)$$

Outerjoin

- $R \dot{\bowtie} S$
- Performs a natural join, but retains the dangling tuples
  - Inserts “NULL” values for dangling tuples
  - Null is designated with this symbol: $\perp$

Example

$PlaysIn$

Band	First	Last
Killers	Amy	Cox
Nails	Billy	Day
Loud	Kevin	Smith

$Plays$

First	Last	Instrument
Amy	Cox	Guitar
Amy	Cox	Vocals
Jeff	Gill	Flute

$PlaysIn \;\dot{\bowtie}\; Plays$

Band	First	Last	Instrument
Killers	Amy	Cox	Guitar
Killers	Amy	Cox	Vocals
Nails	Billy	Day	$\perp$
Loud	Kevin	Smith	$\perp$
$\perp$	Jeff	Gill	Flute

Outerjoin

- Variants
  - Left outerjoin includes only dangling tuples from the relation on the left hand side
  - Right outerjoin includes only dangling tuples from the relation on the right hand side
- All variants have theta equivalent

Example

$PlaysIn$

Band	First	Last
Killers	Amy	Cox
Nails	Billy	Day
Loud	Kevin	Smith

$Plays$

First	Last	Instrument
Amy	Cox	Guitar
Amy	Cox	Vocals
Jeff	Gill	Flute

$PlaysIn \ \dot{\bowtie}_L \ Plays$

Band	First	Last	Instrument
Killers	Amy	Cox	Guitar
Killers	Amy	Cox	Vocals
Nails	Billy	Day	⊥
Loud	Kevin	Smith	⊥

Example

$PlaysIn$

Band	First	Last
Killers	Amy	Cox
Nails	Billy	Day
Loud	Kevin	Smith

$Plays$

First	Last	Instrument
Amy	Cox	Guitar
Amy	Cox	Vocals
Jeff	Gill	Flute

$PlaysIn \ \dot{\bowtie}_R \ Plays$

Band	First	Last	Instrument
Killers	Amy	Cox	Guitar
Killers	Amy	Cox	Vocals
⊥	Jeff	Gill	Flute

Writing Extended Queries

- Given these relations:

  Album(AlbumTitle,BandName,YearReleased, Price)

  Band(BandName,City,Genre,YearFormed,Label)

Writing Extended Queries

- Write queries for the following
  1. The number of bands in each genre
  2. The price for all the albums by each band
  3. The price for all albums from a given decade
  4. The price for all albums from each genre, including “NULL” total for unknown bands
  5. The longest number of years a band released an album after forming

Example 1

- The number of bands in each genre

$$\gamma_{Genre,COUNT(BandName) \rightarrow bandCount}(Band)$$

Example 2

- The price for all the albums by each band

$$\gamma_{BandName, SUM(Price) \rightarrow (bandTotal)}(Album)$$

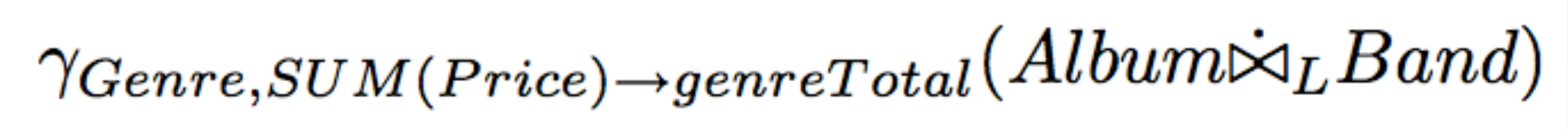
Example 3

- The price for all albums from a given decade

$$\gamma_{decade,SUM(Price)\rightarrow decadeTotal}\left(\pi_{(YearReleased/10)*10\rightarrow decade,Price}(Album)\right)$$

Example 4

- The price for all albums from a genre, including "NULL" for unknown bands



$$\gamma_{Genre, SUM(Price) \rightarrow genreTotal}(Album \overset{\cdot}{\bowtie}_L Band)$$

Example 5

- The longest number of years a band released an album after forming

$$\gamma_{MAX(yearsBetween)}(\pi_{YearReleased-YearFormed\rightarrow yearsBetween}(Band \bowtie Album))$$

Practical implementation

- We have mostly been studying queries in the abstract
- Let’s start learning a practical query language

SQL

- *Structured query language*
- Most common DBMS language
- DML components very similar to extended relational algebra
- Syntax reads like an English sentence

Example query

```
SELECT albumName
FROM Album
WHERE bandName=“Nirvana”;
```

$$\pi_{AlbumName}(\sigma_{BandName="Nirvana"}(Album))$$

Breaking it down

- SELECT - identifies the attributes (columns) to include in the result
  - like “projection” operator
- FROM - identifies the relation (table)
- WHERE - indicates conditions about tuples (rows) to be collected
  - like “select” operator

Projection (SELECT)

- We can indicate multiple attributes
- We can perform computation on the attributes
- We can rename attributes with the AS

Example

```
SELECT albumTitle,
    (yearReleased/10)*10 as decadeRelased,
    price*.8 as discountPrice
FROM Album;
```

Selection (WHERE)

- We can indicate boolean expressions (similar to C)
- <> is the symbol for “Not equal”
- = is the symbol for “equal”

Example

```
SELECT AlbumTitle as cheap80sAlbum
FROM Album
WHERE price<3 AND yearReleased<1990
    AND yearReleased>=1980;
```

Pattern matching

- We can use the keyword LIKE to specify patterns in our conditions
- Two special symbols:
  - _ matches any single character
  - % matches any zero or more characters
- Can be used as any part of a WHERE clause

Example

```
SELECT bandName
FROM Album
WHERE bandName LIKE ‘_ _ _’;
```

results: REM, ABC, TLC, POD

Example

```
SELECT bandName
FROM Album
WHERE bandName LIKE ‘N%i%l%’;
```

results: **Ni**ne Inch Nai**l**s, **N**at**i**ona**l**, **N**eutral M**i**lk Hote**l**

Dates

- We can compare with dates using the standard operators
- Don’t need to worry about internal representation
- DATE, TIME, and TIMESTAMP types available
- Can indicate with strings

Example

```
SELECT albumTitle
FROM Album
WHERE releaseData < ‘2000-01-01’;
```

NULL

- When attribute data is unknown or unavailable for a tuple
- Comparing to NULL values results in UKNOWN (not true, but not false)
- Tuples with UNKNOWN results in WHERE clause are not returned

Sorting

- We can get result of our query sorted using ORDER BY
- Can specify ASC or DESC if we would like the list sorted in ascending or descending order
  - Ascending is the default

Example

```
SELECT *
FROM Song
ORDER BY length DESC;
```

SongTitle	AlbumTitle	Length
Siva	Gish	4:21
Feel The Pain	Without a Sound	4:18
Lithium	Nevermind	4:17
Breed	Nevermind	3:03

Combining relations

- We can specify more than one table in the FROM clause
- This will join tuples of both tables
  - similar to a cartesian product of two relations
- Can use dot operator (period) to refer to specific attributes

Example

```
SELECT Album.albumTitle, bandName, length
FROM Song, Album
WHERE
    Song.albumTitle=Album.albumTitle AND
    Album.yearReleased>=1980 AND
    bandName LIKE ‘N%i%l%’;
```

Album.albumTitle	bandName	length
In the Aeroplane Over the Sea	Neutral Milk Hotel	4:26
High Violet	National	3:25

Example

```
SELECT A1.albumTitle, A2.albumTitle
FROM Album A1, Album A2
Where A1.albumTitle<>A2.albumTitle
```

A1.albumTitle	A1.albumTitle
In the Aeroplane Over the Sea	High Violet
In the Aeroplane Over the Sea	In the Airplane Over the Sea
High Violet	In the Aeroplane Over the Sea
High Violet	Oh, Inverted World
Oh, Inverted World	High Violet
Oh, Inverted World	In the Airplane Over the Sea

Set Operations

- Can be specified between two queries
  - – UNION
  - – INTERCETION
  - – EXCEPT (difference)

Example

```
(SELECT bandName as name FROM Band)
    INTERSECT
(
    (SELECT albumName as name FROM Album)
        UNION
    (Select songName as name FROM Song)
);
```

name
Wilco