CSE 444: Database Internals

Lecture 2
Review of the
Relational Model and SQL

Magda Balazinska - CSE 444, Spring 2011

Relation Definition

- · Database is collection of relations
- Relation R is subset of S₁ x S₂ x ... x S_n
 - Where S_i is the domain of attribute i
 - **n** is number of attributes of the relation
- · Relation is basically a table with rows & columns
 - SQL uses word table to refer to relations

Magda Balazinska - CSE 444, Spring 2011

2

Properties of a Relation

- · Each row represents an n-tuple of R
- · Ordering of rows is immaterial
- · All rows are distinct
- · Ordering of columns is significant
 - Because two columns can have same domain
 - But columns are labeled so
 - Applications need not worry about order
 - They can simply use the names
- · Domain of each column is a primitive type
- Relation consists of a relation schema and instance

Magda Balazinska - CSE 444, Spring 2011

More Definitions

- · Relation schema: describes column heads
 - Relation name
 - Name of each field (or column, or attribute)
 - Domain of each field
- · Degree (or arity) of relation: nb attributes
- Database schema: set of all relation schemas

Magda Balazinska - CSE 444, Spring 2011

4

Even More Definitions

- Relation instance: concrete table content
 - Set of tuples (also called records) matching the schema
- · Cardinality of relation instance: nb tuples
- Database instance: set of all relation instances

Magda Balazinska - CSE 444, Spring 2011

Example

- Relation schema
 Supplier(sno: integer, sname: string, scity: string, sstate: string)
- Relation instance

| sno | sname | scity | sstate |
|-----|-------|--------|--------|
| 1 | s1 | city 1 | WA |
| 2 | s2 | city 1 | WA |
| 3 | s3 | city 2 | MA |
| 4 | s4 | city 2 | MA |

Magda Balazinska - CSE 444, Spring 2011

Integrity Constraints

- · Integrity constraint
 - Condition specified on a database schema
 - Restricts data that can be stored in db instance
- · DBMS enforces integrity constraints
 - Ensures only legal database instances exist
- · Simplest form of constraint is domain constraint
 - Attribute values must come from attribute domain

Magda Balazinska - CSE 444, Spring 2011

Key Constraints

- · Key constraint: "certain minimal subset of fields is a unique identifier for a tuple"
- · Candidate key
 - Minimal set of fields
 - That uniquely identify each tuple in a relation
- Primary key
 - One candidate key can be selected as primary key

Magda Balazinska - CSE 444, Spring 2011

Foreign Key Constraints

- · A relation can refer to a tuple in another relation
- Foreign key
 - Field that refers to tuples in another relation
 - Typically, this field refers to the primary key of other
 - Can pick another field as well

Magda Balazinska - CSE 444, Spring 2011

Key Constraint SQL Examples

```
CREATE TABLE Part (
 pno integer,
 pname varchar(20),
 psize integer,
 pcolor varchar(20),
  PRIMARY KEY (pno)
```

Magda Balazinska - CSE 444, Spring 2011

10

12

Key Constraint SQL Examples

```
CREATE TABLE Supply(
  sno integer,
  pno integer,
  qty integer,
  price integer
```

Magda Balazinska - CSE 444, Spring 2011

11

Key Constraint SQL Examples

```
CREATE TABLE Supply(
 sno integer,
 pno integer,
 qty integer,
 price integer,
  PRIMARY KEY (sno,pno)
```

Magda Balazinska - CSE 444, Spring 2011

Key Constraint SQL Examples

```
CREATE TABLE Supply(
sno integer,
pno integer,
qty integer,
price integer,
PRIMARY KEY (sno,pno),
FOREIGN KEY (sno) REFERENCES Supplier,
FOREIGN KEY (pno) REFERENCES Part
);

Magda Balazinska-CSE 444, Spring 2011 13
```

Key Constraint SQL Examples

```
CREATE TABLE Supply(
sno integer,
pno integer,
qty integer,
price integer,
PRIMARY KEY (sno,pno),
FOREIGN KEY (sno) REFERENCES Supplier
ON DELETE NO ACTION,
FOREIGN KEY (pno) REFERENCES Part
ON DELETE CASCADE
);

Magda Balazinska - CSE 444. Spring 2011
14
```

General Constraints

 Table constraints serve to express complex constraints over a single table

```
CREATE TABLE Part (
pno integer,
pname varchar(20),
psize integer,
pcolor varchar(20),
PRIMARY KEY (pno),
CHECK ( psize > 0 )
);
```

Note: Also possible to create constraints over many tables

Magda Balazinska - CSE 444, Spring 2011

Relational Queries

- Query inputs and outputs are relations
- · Query evaluation
 - Input: instances of input relations
 - Output: instance of output relation

Magda Balazinska - CSE 444, Spring 2011

16

Relational Algebra

- · Query language associated with relational model
- · Queries specified in an operational manner
 - A query gives a step-by-step procedure
- · Relational operators
 - Take one or two relation instances as argument
 - Return one relation instance as result
 - Easy to compose into relational algebra expressions

Magda Balazinska - CSE 444, Spring 2011

17

Relational Operators

- Selection: $\sigma_{condition}(S)$
 - Condition is Boolean combination (A,V) of terms
 - Term is: attr. op constant, attr. op attr.
 - Op is: <, <=, =, ≠, >=, or >
- Projection: π_{list-of-attributes}(S)
- Union (∪), Intersection (∩), Set difference (–),
- Cross-product or cartesian product (x)
- Join: $R_{\bowtie_{\theta}}S = \sigma_{\theta}(R \times S)$
- Division: R/S, Rename ρ(R(F),E)

Magda Balazinska - CSE 444, Spring 2011

Selection & Projection Examples

| Patient | | | | | | |
|---------|------|-------|---------|--|--|--|
| no | name | zip | disease | | | |
| 1 | p1 | 98125 | flu | | | |
| 2 | p2 | 98125 | heart | | | |
| 3 | р3 | 98120 | lung | | | |
| 4 | p4 | 98120 | heart | | | |

(Patient) $\pi_{zip,\underline{dis}}$

| p,c | disease | | | |
|-----|---------|---------|--|--|
| | zip | disease | | |
| | 98125 | flu | | |
| | 98125 | heart | | |
| | 98120 | lung | | |
| | 98120 | heart | | |

$\sigma_{disease='heart'}(Patient)$

| no | name | zip | disease |
|----|------|-------|---------|
| 2 | p2 | 98125 | heart |
| 4 | p4 | 98120 | heart |

 $\pi_{zip}\left(\sigma_{disease='heart'}(Patient)\right)$

| zip | |
|-------|--|
| 98120 | |
| 98125 | |

Magda Balazinska - CSE 444, Spring 2011

Relational Operators

- Selection: $\sigma_{condition}(S)$
 - Condition is Boolean combination (A,v) of terms
 - Term is: attr. op constant, attr. op attr.
 - Op is: <, <=, =, ≠, >=, or >
- Projection: $\pi_{list\text{-of-attributes}}(S)$
- Union (∪), Intersection (∩), Set difference (–),
- Cross-product or cartesian product (x)
- Join: $R_{\bowtie_{\theta}}S = \sigma_{\theta}(R \times S)$
- Division: R/S, Rename ρ(R(F),E)

Magda Balazinska - CSE 444, Spring 2011

Cross-Product Example

AnonPatient P

| age | zip | disease |
|-----|-------|---------|
| 54 | 98125 | heart |
| 20 | 98120 | flu |

Voters V

| name | age | zip |
|------|-----|-------|
| p1 | 54 | 98125 |
| p2 | 20 | 98120 |

$P{\bowtie}V$

| P.age | P.zip | disease | name | V.age | V.zip |
|-------|-------|---------|------|-------|-------|
| 54 | 98125 | heart | p1 | 54 | 98125 |
| 54 | 98125 | heart | p2 | 20 | 98120 |
| 20 | 98120 | flu | p1 | 54 | 98125 |
| 20 | 98120 | flu | p2 | 20 | 98120 |

Magda Balazinska - CSE 444, Spring 2011

23

Different Types of Join

- Theta-join: $R \bowtie_{\theta} S = \sigma_{\theta}(R \times S)$
 - Join of R and S with a join condition $\boldsymbol{\theta}$
 - Cross-product followed by selection $\boldsymbol{\theta}$
- Equijoin: $R_{\bowtie_{\theta}}S = \pi_A (\sigma_{\theta}(R \times S))$
 - Join condition θ consists only of equalities
 - Projection π_A drops all redundant attributes
- Natural join: $R_{\bowtie} S = \pi_A (\sigma_{\theta}(R \times S))$
 - Equijoin
 - Equality on **all** fields with same name in R and in S

Magda Balazinska - CSE 444, Spring 2011

22

Theta-Join Example

AnonPatient P

| age | zip | disease |
|-----|-------|---------|
| 54 | 98125 | heart |
| 20 | 98120 | flu |

Voters V

| name | age | zip |
|------|-----|-------|
| p1 | 54 | 98125 |
| p2 | 20 | 98120 |

P⊠_{Page=Vage ∧ Pzip=Azip ∧ Page < 50} V

| | F.age - V.age A F.zip - A.zip A F.age < 50 | | | | | | |
|---|--|-------|---------|------|-------|-------|--|
| | P.age | P.zip | disease | name | V.age | V.zip | |
| ı | 20 | 98120 | flu | n2 | 20 | 98120 | |

Magda Balazinska - CSE 444, Spring 2011

Equijoin Example

AnonPatient P

| Anoniralient r | | | | | |
|----------------|-------|---------|--|--|--|
| age | zip | disease | | | |
| 54 | 98125 | heart | | | |
| 20 | 98120 | flu | | | |

| Voters V | | |
|----------|-----|-------|
| name | age | zip |
| p1 | 54 | 98125 |
| n2 | 20 | 98120 |

P⊠_{Pane=Vane} V

| r.age=v.age | | | | |
|-------------|-------|---------|------|-------|
| age | P.zip | disease | name | V.zip |
| 54 | 98125 | heart | p1 | 98125 |
| 20 | 98120 | flu | p2 | 98120 |

Magda Balazinska - CSE 444, Spring 2011

Natural Join Example

AnonPatient P

| age | zip | disease | |
|-----|-------|---------|--|
| 54 | 98125 | heart | |
| 20 | 98120 | flu | |

Voters V

| name | age | zip |
|------|-----|-------|
| p1 | 54 | 98125 |
| p2 | 20 | 98120 |

$P \bowtie V$

| age | zip | disease | name |
|-----|-------|---------|------|
| 54 | 98125 | heart | p1 |
| 20 | 98120 | flu | p2 |

Magda Balazinska - CSE 444, Spring 2011

More Joins

- · Outer join
 - Include tuples with no matches in the output
 - Use NULL values for missing attributes
- Variants
 - Left outer join
 - Right outer join
 - Full outer join

Magda Balazinska - CSE 444, Spring 2011

26

Outer Join Example

AnonPatient P

| age | zip | disease |
|-----|-------|---------|
| 54 | 98125 | heart |
| 20 | 98120 | flu |
| 33 | 98120 | luna |

Voters V

| VOICES V | | | |
|----------|-----|-------|--|
| name | age | zip | |
| p1 | 54 | 98125 | |
| p2 | 20 | 98120 | |

P⊗V

| age | zip | disease | name |
|-----|-------|---------|------|
| 54 | 98125 | heart | p1 |
| 20 | 98120 | flu | p2 |
| 33 | 98120 | lung | null |

Magda Balazinska - CSE 444, Spring 2011

Example of Algebra Queries

Q1: Names of patients who have heart disease $\pi_{\text{name}}(\text{Voter}\bowtie(\sigma_{\text{disease='heart'}}(\text{AnonPatient}))$

Magda Balazinska - CSE 444, Spring 2011

28

More Examples

Relations

Supplier(sno, sname, scity, sstate)
Part(pno, pname, psize, pcolor)
Supply(sno, pno, qty, price)

Q2: Name of supplier of parts with size greater than 10 $\pi_{\rm sname}({\rm Supplier}_{\bowtie}{\rm Supply}_{\bowtie}(\sigma_{\rm psize>10}~({\rm Part}))$

Q3: Name of supplier of red parts or parts with size greater than 10 $\pi_{\text{sname}}(\text{Supplier}_{\bowtie}\text{Supply}_{\bowtie}(\sigma_{\text{psize}>10} \, (\text{Part}) \cup \sigma_{\text{pcolor=red'}} \, (\text{Part}) \,) \,)$

(Many more examples in the book)

Magda Balazinska - CSE 444, Spring 2011

29

27

Extended Operators of Relational Algebra

- Duplicate elimination (δ)
 - Since commercial DBMSs operate on multisets not sets
- Aggregate operators (γ)
 - Min, max, sum, average, count
- Grouping operators (γ)
 - Partitions tuples of a relation into "groups"
 - Aggregates can then be applied to groups
- Sort operator (τ)

Magda Balazinska - CSE 444, Spring 2011

Structured Query Language: SQL

- Influenced by relational calculus (see 344)
- · Declarative query language
- · Multiple aspects of the language
 - Data definition language
 - · Statements to create, modify tables and views
 - Data manipulation language
 - · Statements to issue queries, insert, delete data
 - More

Magda Balazinska - CSE 444, Spring 2011

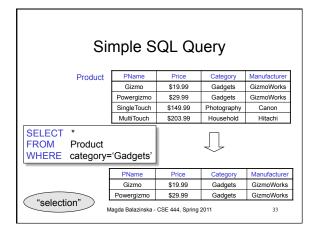
31

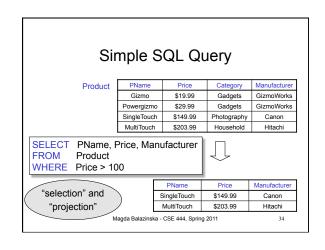
SQL Query

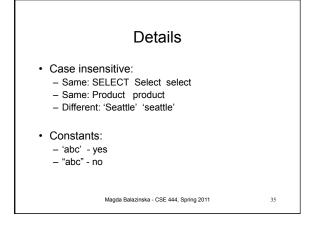
Basic form: (plus many many more bells and whistles)

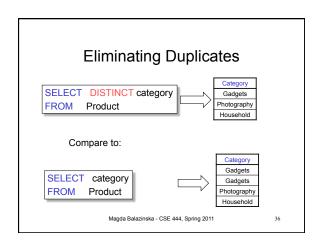
SELECT <attributes>
FROM <one or more relations>
WHERE <conditions>

Magda Balazinska - CSE 444, Spring 2011 32









Ordering the Results

SELECT pname, price, manufacturer
FROM Product
WHERE category='gizmo' AND price > 50

ORDER BY price, pname

Ties are broken by the second attribute on the ORDER BY list, etc.

Ordering is ascending, unless you specify the DESC keyword.

Magda Balazinska - CSE 444, Spring 2011

Joins

Product (<u>pname</u>, price, category, manufacturer) Company (<u>cname</u>, stockPrice, country)

Find all products under \$200 manufactured in Japan; return their names and prices.

SELECT PName, Price

FROM Product, Company

WHERE Manufacturer=CName AND Country='Japan'

AND Price <= 200

Magda Balazinska - CSE 444, Spring 2011

2

Tuple Variables Person(pname, address, worksfor) Company(cname, address) Which address? SELECT DISTINCT pname, address FROM Person, Company WHERE worksfor = cname **DISTINCT** Person.pname, Company.address SELECT FROM Person, Company WHERE Person.worksfor = Company.cname **SELECT** DISTINCT x.pname, y.address **FROM** Person AS x, Company AS y

WHERE x.worksfor = y.cname

Nested Queries

- · Nested query
 - Query that has another query embedded within it
 - The embedded query is called a subquery
- · Why do we need them?
 - Enables to refer to a table that must itself be computed
- · Subqueries can appear in
 - WHERE clause (common)
 - FROM clause (less common)
 - HAVING clause (less common)

Magda Balazinska - CSE 444, Spring 2011

40

Subqueries Returning Relations

Company(<u>name</u>, city) Product(<u>pname</u>, maker) Purchase(<u>id</u>, product, buyer)

Return cities where one can find companies that manufacture products bought by Joe Blow

SELECT Company.city FROM Company

WHERE Company.name IN

(SELECT Product.maker FROM Purchase, Product

WHERE Product.pname=Purchase.product
AND Purchase .buyer = 'Joe Blow');

39

Subqueries Returning Relations

You can also use: s > ALL R

s > ANY R EXISTS R

Product (pname, price, category, maker)

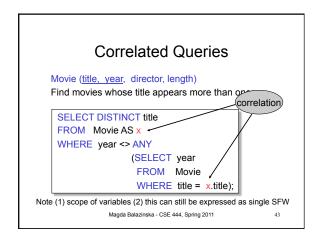
Find products that are more expensive than all those produced By "Gizmo-Works"

SELECT name FROM Product

WHERE price > ALL (SELECT price

FROM Purchase

WHERE maker='Gizmo-Works')





SELECT avg(price)
FROM Product
WHERE maker="Toyota"

SELECT count(*)
FROM Product
WHERE year > 1995

SQL supports several aggregation operations: sum, count, min, max, avg

Except count, all aggregations apply to a single attribute

Magda Balazinska - CSE 444, Spring 2011

Grouping and Aggregation

SELECT S FROM R₁,...,R_n WHERE C1 GROUP BY a₁,...,a_k HAVING C2

Conceptual evaluation steps:

- Evaluate FROM-WHERE, apply condition C1
- 2. Group by the attributes $a_1,...,a_k$
- 3. Apply condition C2 to each group (may have aggregates)
- 4. Compute aggregates in S and return the result

Read more about it in the book...

Magda Balazinska - CSE 444, Spring 2011