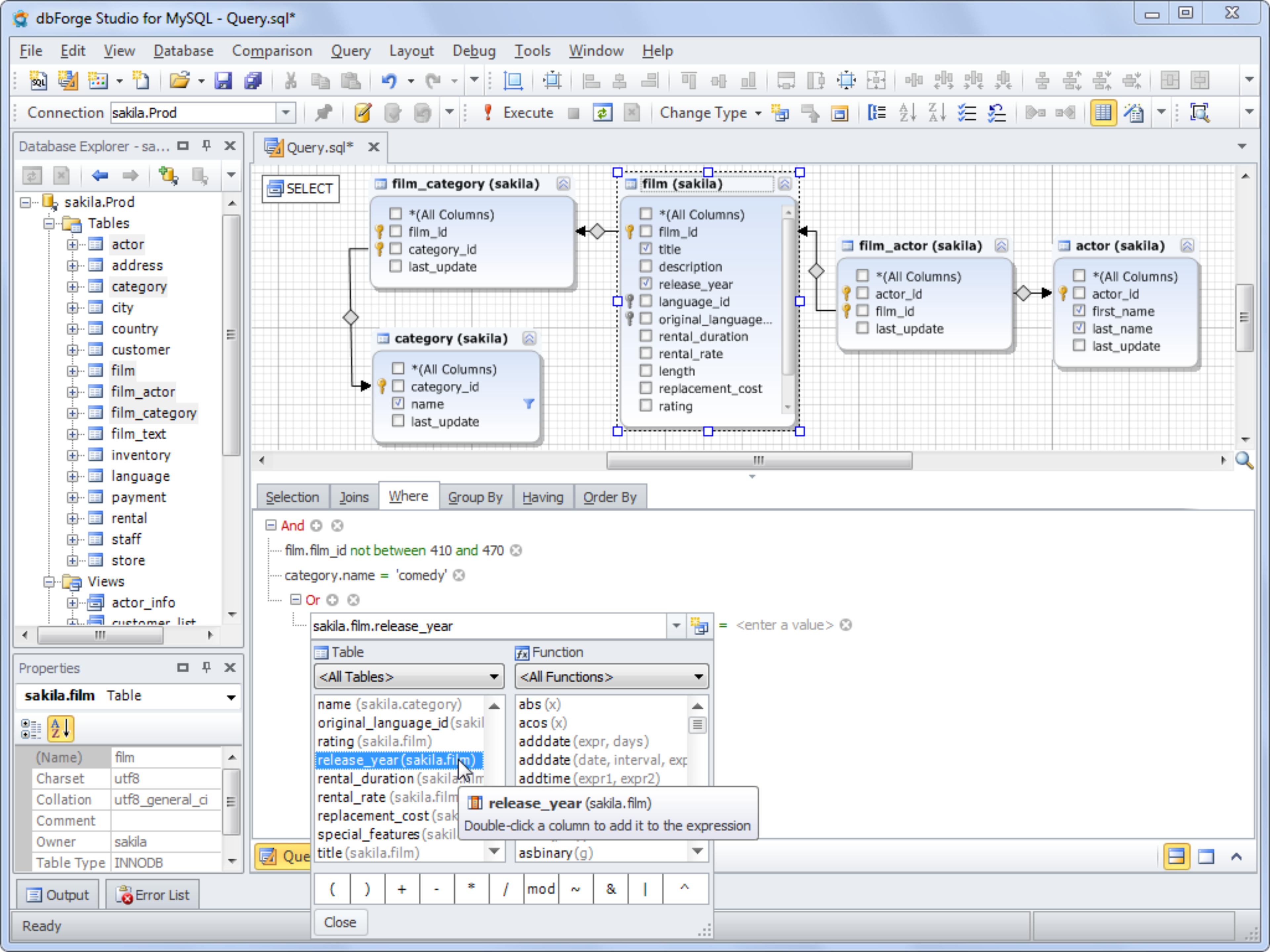
Other Query Interfaces and Languages

Davood Rafiei



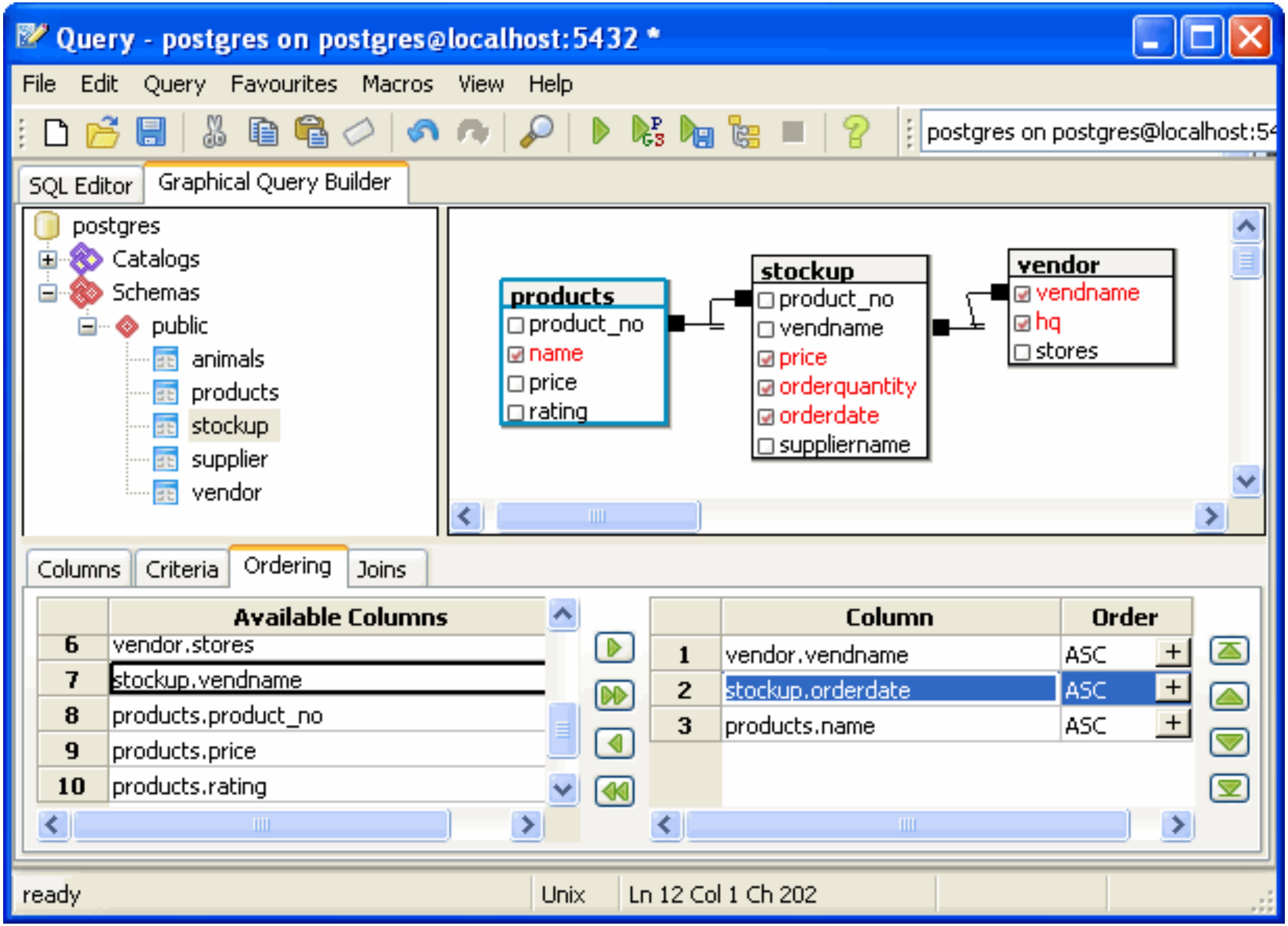
Connection: sakila.Prod

☐ And
film.film_id not between 410 and 470
category.name = 'comedy'
☐ Or
sakila.film.release_year = <enter a value>

release_year (sakila.film)
Double-click a column to add it to the expression

sakila.film Table

(Name)	film
Charset	utf8
Collation	utf8_general_ci
Comment	
Owner	sakila
Table Type	INNODB



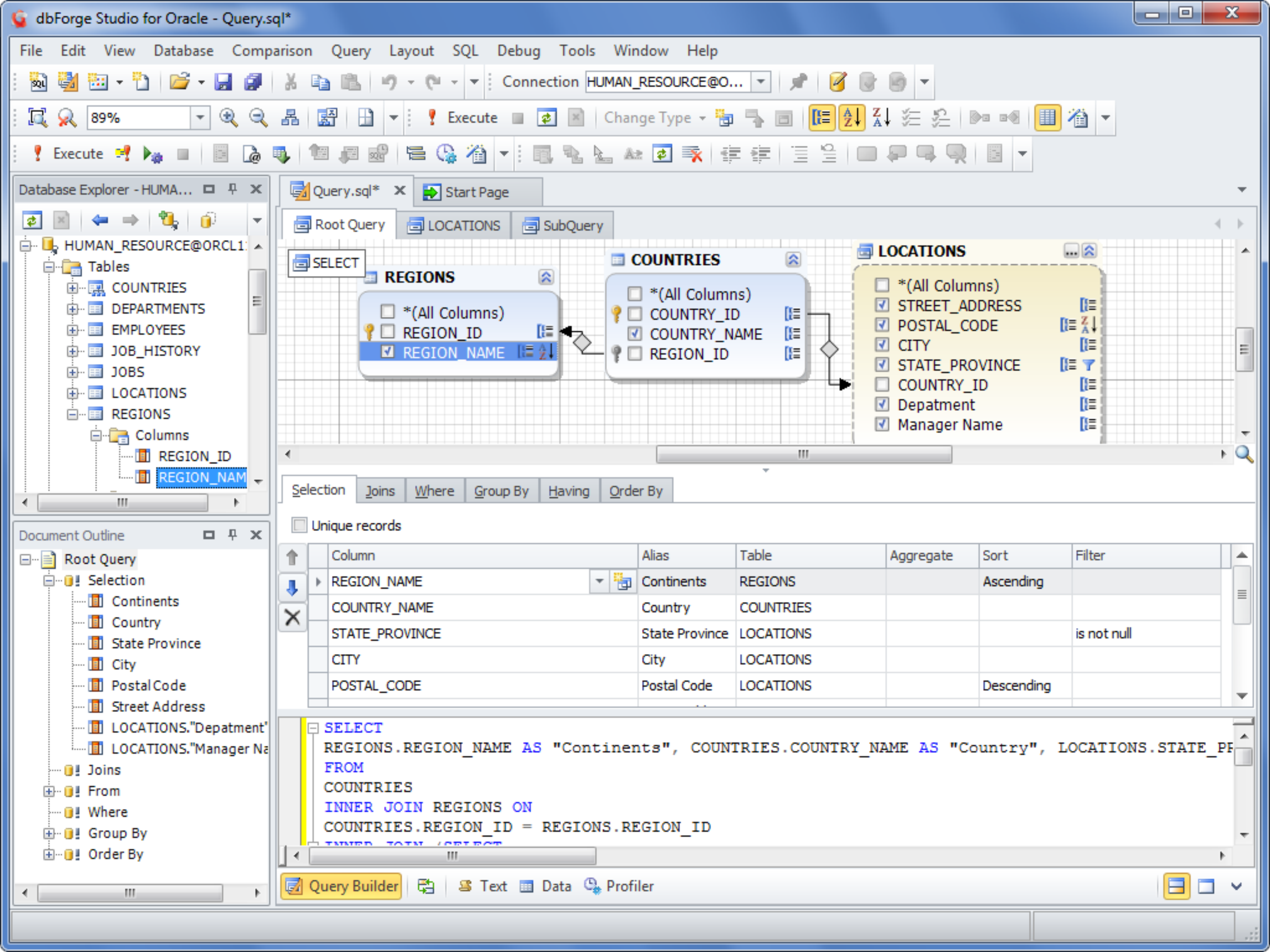
products
- ☐ product_no
- ☑ name
- ☐ price
- ☐ rating

stockup
- ☐ product_no
- ☐ vendname
- ☑ price
- ☑ orderquantity
- ☑ orderdate
- ☐ suppliername

vendor
- ☑ vendname
- ☑ hq
- ☐ stores

	Available Columns
6	vendor.stores
7	stockup.vendname
8	products.product_no
9	products.price
10	products.rating

	Column	Order
1	vendor.vendname	ASC
2	stockup.orderdate	ASC
3	products.name	ASC



☐ Unique records

Column	Alias	Table	Aggregate	Sort	Filter
REGION_NAME	Continents	REGIONS		Ascending	
COUNTRY_NAME	Country	COUNTRIES			
STATE_PROVINCE	State Province	LOCATIONS			is not null
CITY	City	LOCATIONS			
POSTAL_CODE	Postal Code	LOCATIONS		Descending	

```
SELECT
REGIONS.REGION_NAME AS "Continents", COUNTRIES.COUNTRY_NAME AS "Country", LOCATIONS.STATE_PF
FROM
COUNTRIES
INNER JOIN REGIONS ON
COUNTRIES.REGION_ID = REGIONS.REGION_ID
INNER JOIN (SELECT
```

Graphical Query Interfaces

- Oracle SQL query builder
- MySQL query builder
- SQL Server visual query builder
- Postgress query builder
- SQLite query builder
- Microsoft Access query interface

Roots

- Based on relational calculus
  - So is SQL
- Declarative
- Known as Query by Example (QBE)
- Limited expressive power

QBE: Query by Example

- Declarative query language, like SQL
- Developed in 1970s at IBM
- Based on DRC
- Visual
- Other visual query languages (MS Access, Paradox) are just incremental improvements

QBE - Examples

Q1. Print all professors' names in the Math department

Professor	*Id*	*Name*	*DeptId*
		P.	Math

Q2. Print all professors' names who taught c291 in Fall 2002.

Professor	*Id*	*Name*	*DeptId*
	_123	**P.**	

Teaching	*ProfId*	*CrsCode*	*Semester*
	_123	C291	F2002

Condition Boxes

- Some conditions are too complex to be placed directly in table columns

Transcript	*StudId*	*CrsCode*	*Semester*	*Grade*
	P.	CS532		_Gr

Conditions
_Gr = 'A' OR _Gr = 'B'

- Students who took CS532 & got A or B

Connection to Rel. Calculus

- A graphical representation of DRC

Transcript	*StudId*	*CrsCode*	*Semester*	*Grade*
	_123	_CS532	F2002	A

DRC: Transcript(*x*, *y*, ‘F2002’, ‘A’)

TRC: Transcript(*t*) AND t.Semester=‘F2002’ AND t.Grade = ‘A’

Relational Calculus

- Two flavors
  - Tuple Relational Calculus (TRC)
    - Variables range over tuples (e.g. SQL)
  - Domain Relational Calculus (DRC)
    - Variables range over domains (e.g. QBE)
- Query ~ formula
- Answer ~ an assignment that makes the formula true

Examples

TRC:

{t | Transcript(*t*) AND t.Semester=‘F2002’ AND t.Grade = ‘A’}

{s | Students(s) AND ∃e∈Enroll (s.sid = e.sid AND e.cid=‘291’)}

{s | Students(s) AND ∃c∈Courses (c.instructor=‘John Smith’
==> ∃e∈Enroll (e.sid = s.sid AND e.cid=c.cid))}

DRC:

{x,y | Transcript(*x*, *y*, ‘F2002’, ‘A’)}

Relational Calculus

a base for other query languages

Relation Between TRC and SQL

- List the names of all professors who have taught MGT123
  - In TRC:

$$\{p.\mathit{Name} \mid \text{Professor}(p) \text{ AND } \exists t \in \text{Teaching}\ (p.\mathit{Id} = t.\mathit{ProfId} \text{ AND } t.\mathit{CrsCode} = \text{‘MGT123’})\ \}$$

  - In SQL:

```
SELECT   p.Name
FROM     Professor p, Teaching t
WHERE    p.Id = t.ProfId  AND  t.CrsCode = ‘MGT123’
```

Core of SQL is merely a syntactic sugar on top of TRC

What Happened to Quantifiers in SQL?

- SQL has no quantifiers: how come? It uses conventions:
  - *Convention* 1. Universal quantifiers are not allowed (but SQL:1999 introduced a limited form of explicit ∀)
  - *Convention* 2. Make existential quantifiers *implicit*: Any tuple variable that does not occur in SELECT is assumed to be implicitly quantified with ∃
- Compare:

$$\{p.\mathit{Name} \mid \text{Professor}(p)\ \ \text{AND}\ \ \exists t \in \text{Teaching}\ \ \ldots \}$$

and

```
SELECT    P.Name
FROM      Professor p,  Teaching t
 … … …
```

Implicit
∃

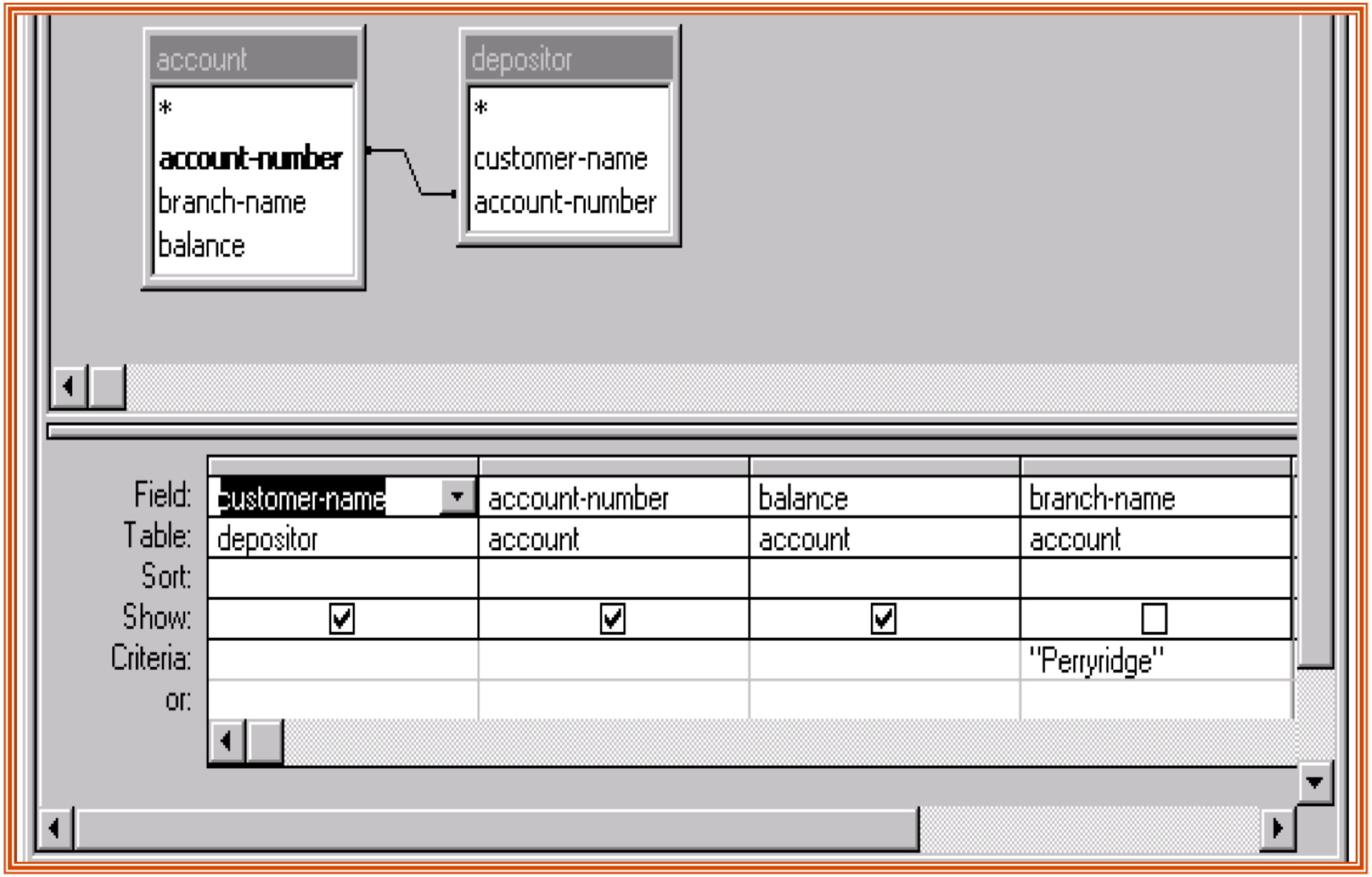
Relation Between TRC and SQL (cont’d)

- SQL uses a subset of TRC with simplifying conventions for quantification
- Restricts the use of quantification and negation (so TRC is more general in this respect)
- SQL uses aggregates, which are absent in TRC (and relational algebra, for that matter). But aggregates can be added
- SQL is extended with relational algebra operators (MINUS, UNION, JOIN, etc.)
  - This is just more syntactic sugar, but it makes queries easier to write

Graphical Interfaces

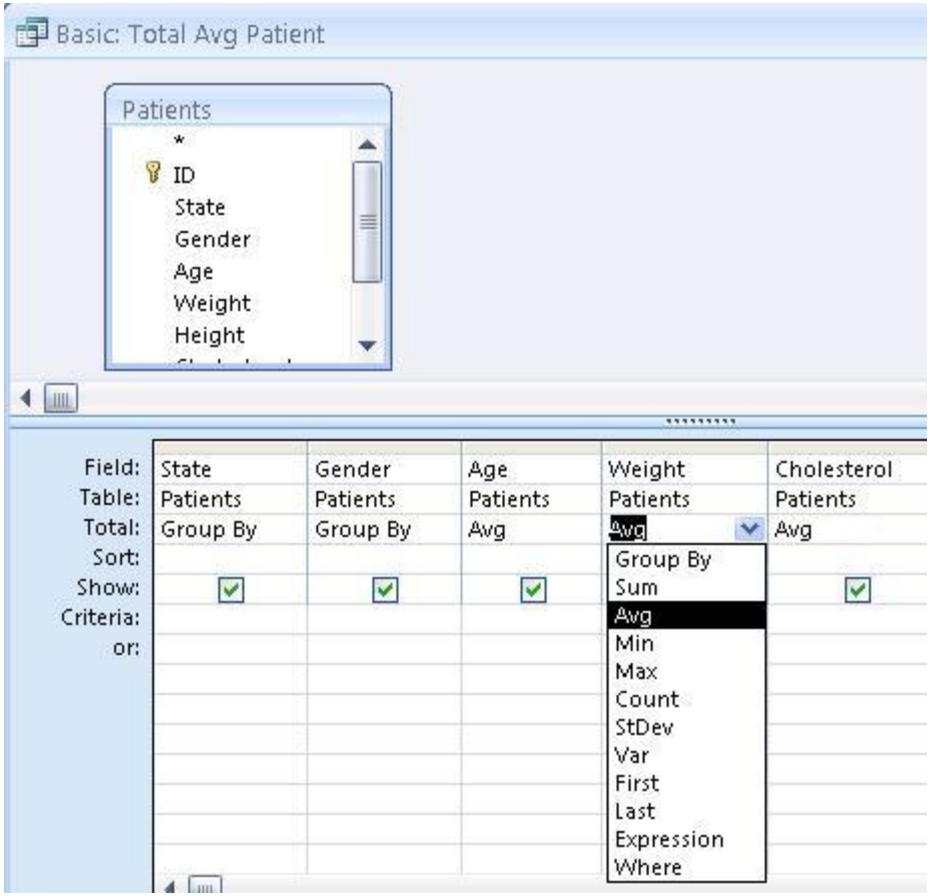
Of PC Databases

Microsoft Access



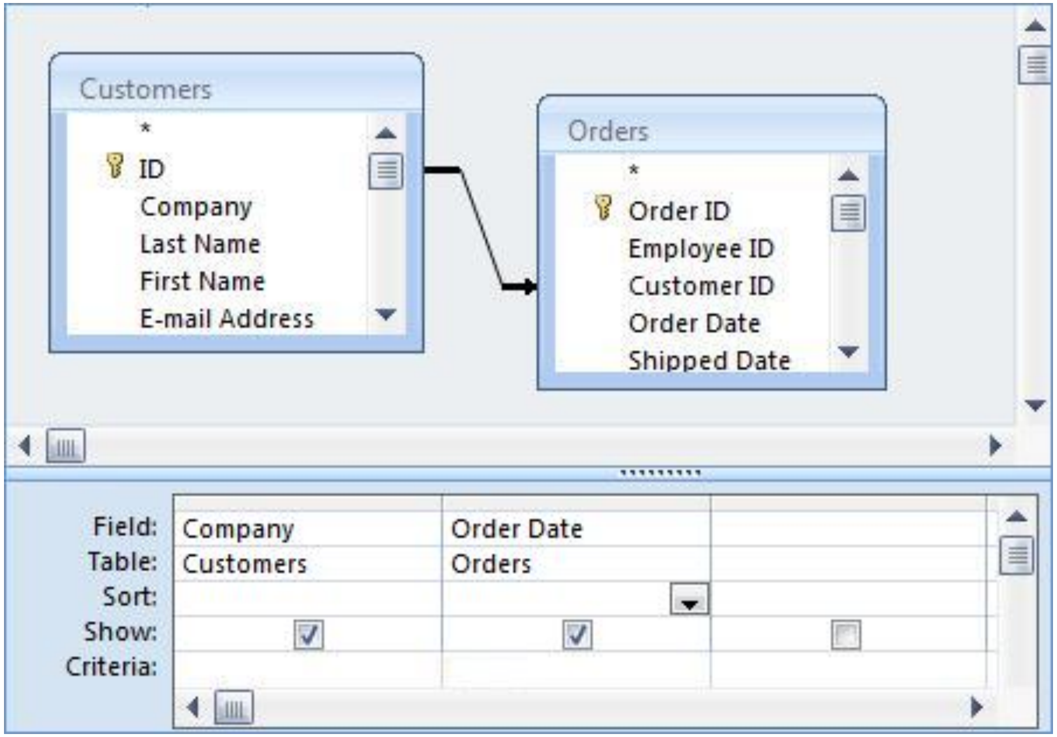
Field:	customer-name	account-number	balance	branch-name
Table:	depositor	account	account	account
Sort:				
Show:	☑	☑	☑	□
Criteria:				"Perryridge"
or:				

Microsoft Access (Cont)



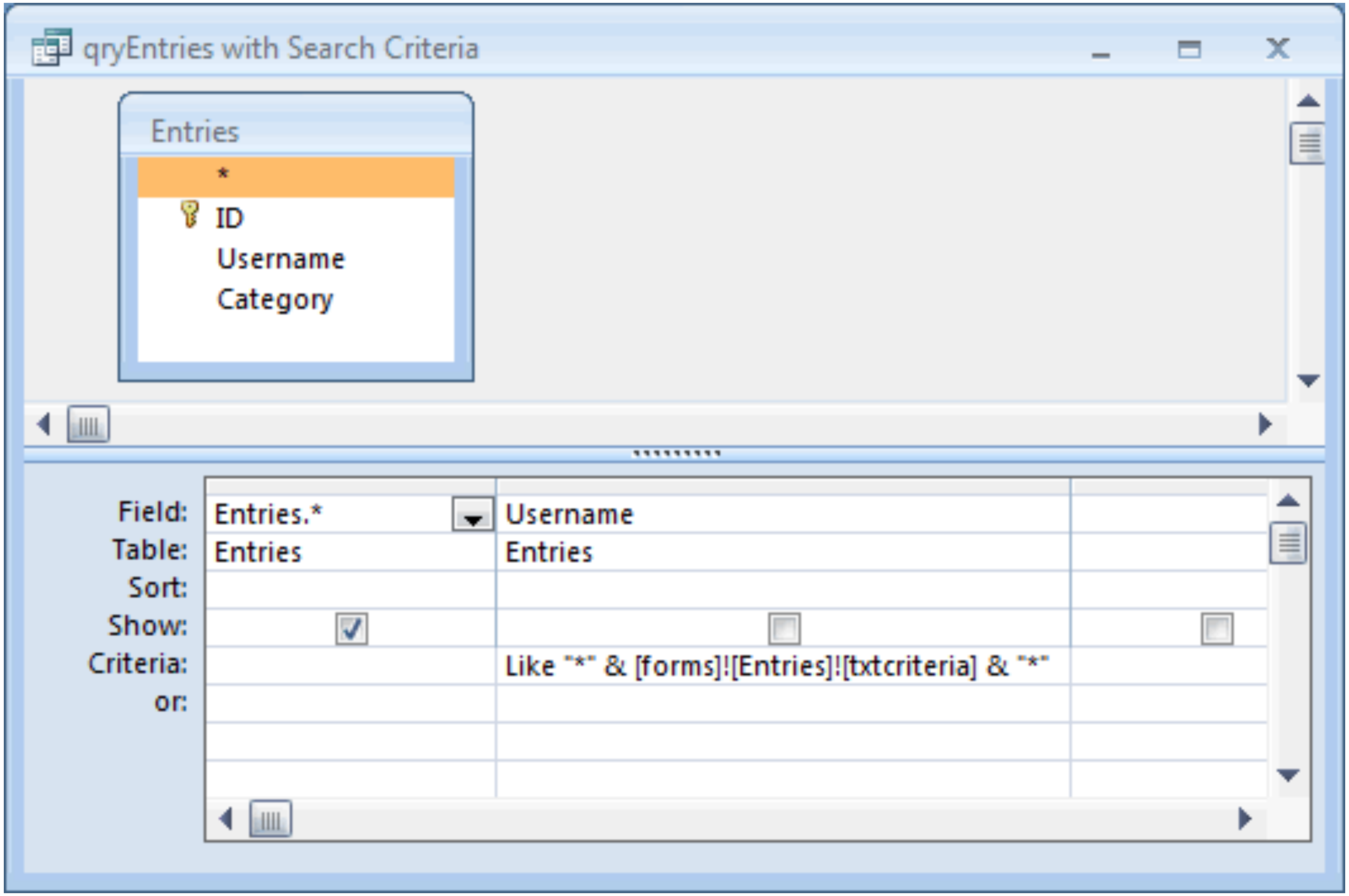
Field:	State	Gender	Age	Weight	Cholesterol
Table:	Patients	Patients	Patients	Patients	Patients
Total:	Group By	Group By	Avg	Avg	Avg
Sort:					
Show:	☑	☑	☑		☑
Criteria:					
or:					

Microsoft Access (Cont)



Field:	Company	Order Date	
Table:	Customers	Orders	
Sort:			
Show:	☑	☑	□
Criteria:			

Microsoft Access (Cont)



qryEntries with Search Criteria

Entries

- *
- ID
- Username
- Category

Field:	Entries.*	Username	
Table:	Entries	Entries	
Sort:			
Show:	☑	□	□
Criteria:		Like "*" & [forms]![Entries]![txtcriteria] & "*"	
or:			

PC Databases

- A spruced up version of QBE (better interface)
- Be aware of implicit quantification
- Beware of negation pitfalls
  – Sec. 13.4 gives some of the pitfalls under the heading “the price of free lunch”