Introduction to Databases

Exercises: Query-By-Example

QBE basics (1)

- A user writes queries by creating **example tables.**
- QBE uses **domain variables**, as in the DRC, to create example tables.
- The domain of a variable is determined by the column in which it appears, and **variable symbols** are prefixed with **underscore (_)** to distinguish them from constants.
- The fields that should appear in the answer are specified by using the command **P.**, which stands for **print**.
    - To print the names and ages of all sailors, we would create the following example table:

Sailors	*sid*	*sname*	*rating*	*age*
		P._N		P._A

- Query in DRC: $\{\langle N,A\rangle \mid \exists\, \mathrm{I,N,T,A}(\langle \mathrm{I,N,T,A}\rangle \in \mathrm{Sailors})\}$

QBE basics (2)

- If we want to print **all fields** in some relation, we can **place P. under the name of the relation**.
  - This notation is like the SELECT * convention in SQL. It is equivalent to placing a P. in every field:

Sailors	*sid*	*sname*	*rating*	*age*
P.				

- Selections are expressed by placing a constant in some field:

Sailors	*sid*	*sname*	*rating*	*age*
P.			10	

  - Placing a constant, say 10, in a column is the same as placing the condition =10.
    We can use other comparison operations (<, >, <=, >=, ¬) as well. For example, we could say < 10 to retrieve sailors with a rating less than 10.

QBE basics (3)

- We can explicitly specify whether **duplicate** tuples in the answer are to be **eliminated** (or ***not***) by putting **UNQ**. (respectively ***ALL***.) under the relation name.
- We can **order** the presentation of the answers through the use of the **AO.** (for **ascending order**) and **DO.** commands **in conjunction with P.** .
  An **optional integer argument** allows us to **sort on more than one field**.
  - For example, we can display the names, ages, and ratings of all sailors in ascending order by age, and for each age, in ascending order by rating as follows:

Sailors	***sid***	***sname***	***rating***	***age***
		P.	AO(2).	AO(1).

Queries over multiple relations (1)

- To find **sailors with a reservation**, we have to combine information from the Sailors and the Reserves relations.
  In particular we have to **select tuples** from the **two relations with the same value** in the join **column *sid***. We do this by **placing the same variable in the *sid* columns** of the two example relations.

Sailors	*sid*	*sname*	*rating*	*age*
	_Id	P._S		

Reserves	sid	bid	day
	_Id		

Queries over multiple relations (2)

- To find sailors who have reserved a boat for **8/24/96** and who are **older than 25**, we could write:

Sailors	*sid*	*sname*	*rating*	*age*
	_Id	P._S		> 25

Reserves	sid	bid	day
	_Id		'8/24/96'

- The following query prints the names and ages of sailors who have reserved some boat that is also reserved by the sailor with id 22:

Sailors	*sid*	*sname*	*rating*	*age*
	_Id	P._N		P._A

Reserves	sid	bid	day
	_Id	_B	
	22	_B	

Queries over multiple relations (3)

- Find the colors of Interlake boats reserved by sailors who have reserved a boat for 8/24/96 and who are older than 25:

Sailors	*sid*	*sname*	*rating*	*age*
	_Id			> 25

Reserves	sid	bid	day
	_Id	_B	‘8/24/96‘

Boats	bid	bname	color
	_B	Interlake	P.

Negation

- We can print the names of sailors who **do not** have a reservation by using the ¬ command in the relation name column:

Sailors	*sid*	*sname*	*rating*	*age*
	_Id	P._S		

Reserves	sid	bid	day
¬	_Id		

Aggregates

- Like SQL, QBE supports the aggregate operations **AVG.**, **COUNT.**, **MAX.**, **MIN.**, and **SUM.**
  - By default, these aggregate operators **do not eliminate duplicates**, with the exception of COUNT., which does eliminate duplicates.
  - To eliminate duplicate values, the variants AVG.UNQ. and SUM.UNQ. must be used.
  - Curiously, there is no variant of COUNT. that does not eliminate duplicates.
- The following query prints the average age of sailors:

Sailors	*sid*	*sname*	*rating*	*age*	
				_A	P.AVG._A

- QBE supports **grouping** through the use of the **G.** command. To print average ages by rating, we could use:

Sailors	*sid*	*Sname*	*rating*	*age*	
			G.	_A	P.AVG._A

Conditions box

- Simple conditions can be expressed directly in columns of the example tables. For more complex conditions QBE provides a feature called a **conditions box**. Conditions boxes are used to do the following:
  - *Express a condition involving two or more columns*, such as _R/_A > 0.2
  - *Express a condition involving an aggregate operation on a group*, for example, AVG. A > 30. In conjunction with G., only columns with either G. or an aggregate operation can be printed!
    - The following query prints those ratings for which the average age is more than 30:

Sailors	*sid*	*sname*	*rating*	*age*
			G.P.	_A

Conditions
AVG._A > 30

  - *Express conditions involving the **AND** and **OR** operators*.
    - We can print the names of sailors who are younger than 20 or older than 30 as follows:

Sailors	*sid*	*sname*	*rating*	*age*
		P.		_A

Conditions
_A < 20 OR 30 < _A

AND/OR queries

- AND and OR can be expressed in QBE **without** using a **conditions box**.
  - We can print the names of sailors who are younger than 30 or older than 20 by simply creating two example rows:

Sailors	*sid*	*sname*	*rating*	*age*
		P.		< 30
		P.		> 20

  - To print the names of sailors who are **both** younger than 30 and older than 20, we use the same variable in the key fields of both rows:

Sailors	*sid*	*sname*	*rating*	*age*
	_Id	P.		< 30
	_Id			> 20

Unnamed columns

- If we want to display some information **in addition** to fields retrieved from a relation, we can create unnamed columns for display.
  - We can print the name of each sailor along with the ratio rating/age as follows:

Sailors	*sid*	*sname*	*rating*	*age*	
		P.	_R	_A	P._R / _A

- All our examples have included P. commands in exactly **one** table. This is a **QBE restriction**. If we want to display fields from more than one table, we have to use unnamed columns.
  - To print the names of sailors along with the dates on which they have a boat reserved, we could use the following:

Sailors	*sid*	*sname*	*rating*	*age*	
	_Id	P.			P._D

Reserves	sid	bid	day
	_Id		_D

Updates - insert

- **Insertion**, **deletion**, and **modification** of a tuple are specified through the commands **I.**, **D.**, and **U.**, respectively.
  - We can **insert a new tuple** into the Sailors relation as follows:

Sailors	*sid*	*sname*	*rating*	*age*
I.	74	Jana	7	41

  - We can insert several tuples, computed essentially through a query, into the Sailors relation as follows:

Sailors	*sid*	*sname*	*rating*	*age*
I.	_Id	_N		_A

Students	*sid*	*name*	*login*	*age*
	_Id	_N		_A

Conditions
_A > 18 OR _N LIKE ‘C%‘

We insert one tuple for each student older than 18 or with a name that begins with C. The rating field of every inserted tuple contains a null value.

Updates - delete

- We can **delete all tuples** with rating > 5 from the Sailors relation as follows:

Sailors	*sid*	*sname*	*rating*	*age*
D.			> 5	

- We can delete all reservations for sailors with rating < 4 by using:

Sailors	*sid*	*sname*	*rating*	*age*
	_Id		< 4	

Reserves	sid	bid	day
D.	_Id		

Updates - modify

- We can **update** the age of the sailor with *sid* 74 to be 42 years by using:

Sailors	*sid*	*sname*	*rating*	*age*
	74			U.42

- The fact that sid is the key is significant here; **we cannot update the key field**, but we can use it to identify the tuple to be modified (in other fields).
  We can also change the age of sailor 74 from 41 to 42 **by incrementing the age value**:

Sailors	*sid*	*sname*	*rating*	*age*
	74			U._A+1

Exercises

QBE

Exercise 1

Consider the following relational schema.

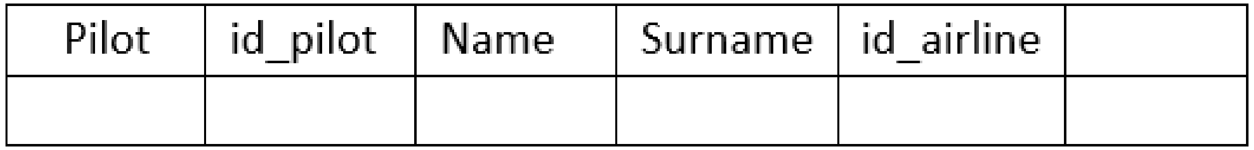
```
aircraft (id_aircraft, id_airline, type, capacity)
airport (id_airport, address, name, country)
airline (id_airline, name, country, address)
pilot (id_pilot, name, surname, id_airline)
flight (id_flight, id_aircraft, id_pilot, id_airport_departure, id_airport_arrival)
```

Write the following queries in QBE.

1. Print all records of pilots, whose surname begin with „B“
2. Print all records of airports that have the penultimate letter of the name „R“ and are not located in Slovenia
3. Print all aircraft types with a capacity of less than 170
4. Print first names and surnames of all pilots who landed in Italy
5. For each pilot, print the name, surname and the name of the airline he or she works for
6. Print airlines that do not own AN-148 (type of the aicraft)

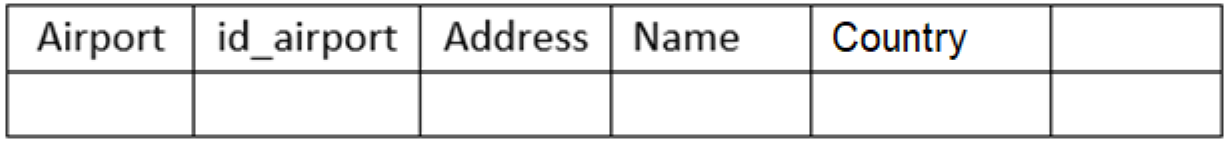
Exercise 1 (1)

1. Print all records of pilots, whose surname begin with „B“



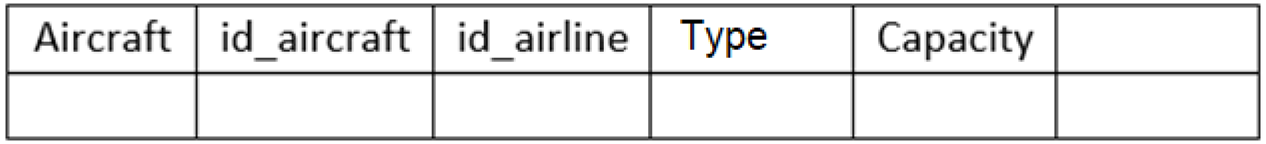
Pilot	id_pilot	Name	Surname	id_airline	

2. Print all records of airports that have the penultimate letter of the name „R“ and are not located in Slovenia



Airport	id_airport	Address	Name	Country	

3. Print all aircrafts types with a capacity of less than 170



Aircraft	id_aircraft	id_airline	Type	Capacity	

Exercise 1 (2)

4. Print first names and surnames of all pilots who landed in Italy

Pilot	id_pilot	Name	Surname	id_airline	

Airport	id_airport	Address	Name	Country	

Flight	id_flight	id_aircraft	id_pilot	id_airport_departure	id_airport_arrival	

5. For each pilot, print the name, surname and the name of the airline he or she works for

Pilot	id_pilot	Name	Surname	id_airline	

Airline	id_airline	Name	Country	address	

6. Print airlines that do not own AN-148 (type of the aicraft)

Airline	id_airline	Name	Country	address	

Aircraft	id_aircraft	id_airline	Type	Capacity	

Exercise 2

Consider the following relational schema. An employee can work in more than one department.

Emp(*eid: integer*, *ename:* string, *salary:* real)

Works(*eid: integer, did: integer*)

Dept(*did:* integer, *dname:* string, *managerid:* integer, *floornum:* integer)

Write the following queries in QBE.

1. Print the names of all employees who work on the 10th floor and make less than $50,000
2. Print the names of all managers who manage three or more departments on the same floor
3. Print the names of all managers who manage ten or more departments on the same floor
4. Give every employee who works in the Toy department a 10% raise
5. Print the names of the departments that employee Santa works in, in ascending order
6. Print the names and salaries of employees who work in both the Toy department and the Candy department
7. Print the names of employees who earn a salary that is either less than $10,000 or more than $100,000
8. Print all of the attributes for employees who work in some department that employee Santa also works in
9. Fire Santa
10. Print the names of employees who make more than 20,000 and work in either the Video department or the Toy department
11. Print the name of each employee who earns more than the manager of the department that he or she works in
12. Print the names of all employees who work on the floor(s) where dwarf John works

Exercise 2 (1)

1. Print the names of all employees who work on the 10th floor and make less than $50,000

Emp	eid	ename	salary	

Works	eid	did	

Dept	did	dname	managerid	floornum	

Conditions

2. Print the names of all managers who manage three or more departments on the same floor

Emp	eid	ename	salary	

Works	eid	did	

Dept	did	dname	managerid	floornum	

Conditions

3. Print the names of all managers who manage ten or more departments on the same floor

Emp	eid	ename	salary	

Works	eid	did	

Dept	did	dname	managerid	floornum	

Conditions

Exercise 2 (2)

4. Give every employee who works in the Toy department a 10% raise

Emp	eid	ename	salary	

Works	eid	did	

Dept	did	dname	managerid	floornum	

Conditions

5. Print the names of the departments that employee Santa works in, in ascending order

Emp	eid	ename	salary	

Works	eid	did	

Dept	did	dname	managerid	floornum	

Conditions

6. Print the names and salaries of employees who work in both the Toy department and the Candy department

Emp	eid	ename	salary	

Works	eid	did	

Dept	did	dname	managerid	floornum	

Conditions

Exercise 2 (3)

7. Print the names of employees who earn a salary that is either less than $10,000 or more than $100,000

Emp	eid	ename	salary	

Works	eid	did	

Dept	did	dname	managerid	floornum	

Conditions

8. Print all of the attributes for employees who work in some department that employee Santa also works in

Emp	eid	ename	salary	

Works	eid	did	

Dept	did	dname	managerid	floornum	

Conditions

9. Fire Santa

Emp	eid	ename	salary	

Works	eid	did	

Dept	did	dname	managerid	floornum	

Conditions

Exercise 2 (4)

10. Print the names of employees who make more than 20,000 and work in either the Video or Toys dept

Emp	eid	ename	salary	

Works	eid	did	

Dept	did	dname	managerid	floornum	

Conditions

11. Print the name of each employee who earns more than the manager of the dept. that he or she works in

Emp	eid	ename	salary	

Works	eid	did	

Dept	did	dname	managerid	floornum	

Conditions

12. Print the names of all employees who work on the floor(s) where dwarf John works

Emp	eid	ename	salary	

Works	eid	did	

Dept	did	dname	managerid	floornum	

Conditions