DATABASE SYSTEMS

Design patterns

Bence Molnár

2022.04.01.

BUDAPESTI MŰSZAKI
ÉS GAZDASÁGTUDOMÁNYI EGYETEM
Építőmérnöki Kar - építőmérnöki képzés 1782 óta

Fotogrammetria és Térinformatika Tanszék

Agenda

- Design patterns
- Practice: flexible table

DESIGN PATTERNS

Design background:

— Own experience

— Rules and regulations

— Design patterns

Design patterns are based on shared experiences and gives guidelines for common situations with generic answers.

— Design patterns are applied in management, building designing etc.

Design patterns

It was first applied by Christopher Alexander in architecture.

- Wikipedia: He was the one looking for patterns that pop up over and over in architecture that characterize well-built houses. In his book, “The Timeless Way of Building,” he tried to describe patterns that could help even a novice architect quickly design good buildings. Due to the many years of experience of the different architects they carried, the designs resulted in more beautiful, better, or more usable houses than if the designer had to design them solely on their own.

- Later, it spread mainly in informatics, including software development: e.g. program design patterns

Design patterns

There are **infinite** number of ways to create a DB.

— Only **some** of them are **optimal**.

— Expertise's' finds them fast.

— For inexperienced it is easier if they get guidelines.

— Here introduced design patters are based on lecturers' experiences and earlier homeworks.

“Report”

Task: A typical problem is when you need to create catalogs and lists and store them in a table. For example, it is often the case that a catalog lists the same manufacturer.

—Solution: To reduce redundancy, these repetitions are collected in a separate table. The key on the code table

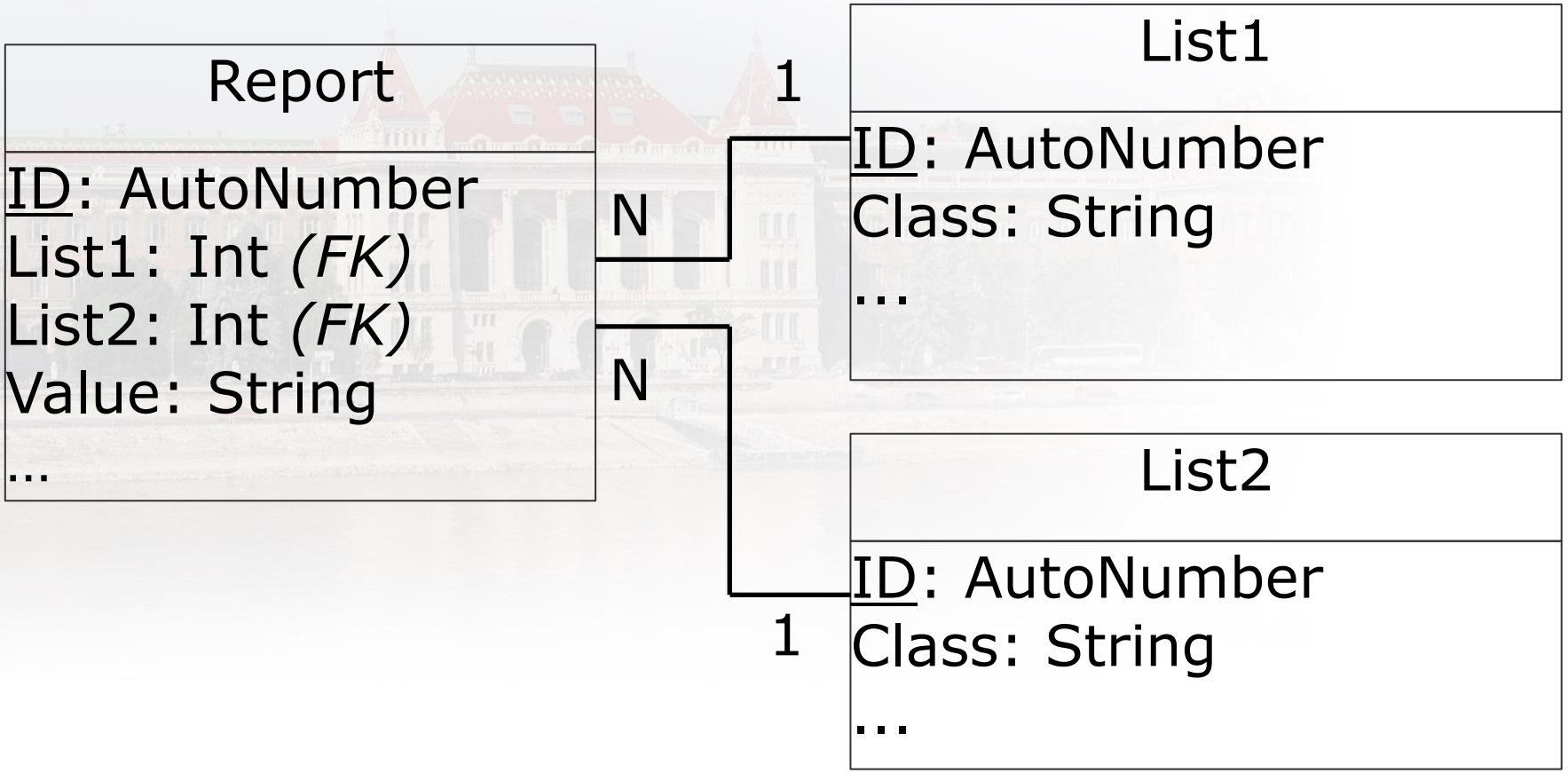
included in the original table as a foreign key.

Example: "Report" – initial state

ID	Name	Type	Manufacturer	Section area	Price
1	T-1	concrete	ConcreteProd Ltd.	1500	150
2	T-2	steel	SteelProd. Ltd.	2500	250
3	P-2	concrete	BigCo Ltd.	4500	450
4	K-1	steel	ConcreteProd Ltd.	1200	650

Mark to recognize: A value in a column occurs multiple time. E.g. concrete or ConcreteProd Ltd.

"Report" - solution



"Report" - solution

ID	Name	TypeID	ManufacturerID	Area	Price
1	T-1	1	1	1500	150
2	T-2	2	2	2500	250
3	P-2	1	3	4500	450
4	K-1	2	1	1200	650

Type

ID	Type	Others
1	Concrete	...
2	Steel	...

Manufacturer

ID	Manufacturer	Address
1	ConsreteProd Ltd.	...
2	SteelProd Ltd.	...
3	BigCo Ltd.	...

“Report” – Pros&Cons

Pros:

- Reducing redundancy
- Easy to add additional info (new columns) to code table (e.g. address column).
- Easy to keep data updated in code table; this enables to keep data clean (e.g. company name).
- Prevents to add duplicate entry to code table by defining key field (e.g. no duplicated manufacturer).

Cons:

- IDs are hardly trackable.
- There is a need to define connection between tables. (e.g. when filtering on a manufacturer’s name)

“Measurement”

Task: Multiple sensors take multiple measurements and record them on a table. We want to record the measurements in a database.

- Solution: Instead of creating a separate table for each sensor, the measurements are stored in one table and the measurements for each instrument are identified by a separate serial number, similarly, the measurements to be treated together are categorized by a different serial number (e.g. simultaneous measurements).

Separated sensors – initial state

Sensor1

MesID: AutoNumber
MesDate: Date
MesValue: Float

Sensor2

MesID: AutoNumber
MesDate: Date
MesValue: Float

Sensor3

MesID: AutoNumber
MesDate: Date
MesValue: Float

Sensor4

MesID: AutoNumber
MesDate: Date
MesValue: Float

Tables have exactly the same attributes!

“Measurement”

Measurement
ID: AutoNumber SensorName: String MesID: Int MesValue: Float …

“Measurement”

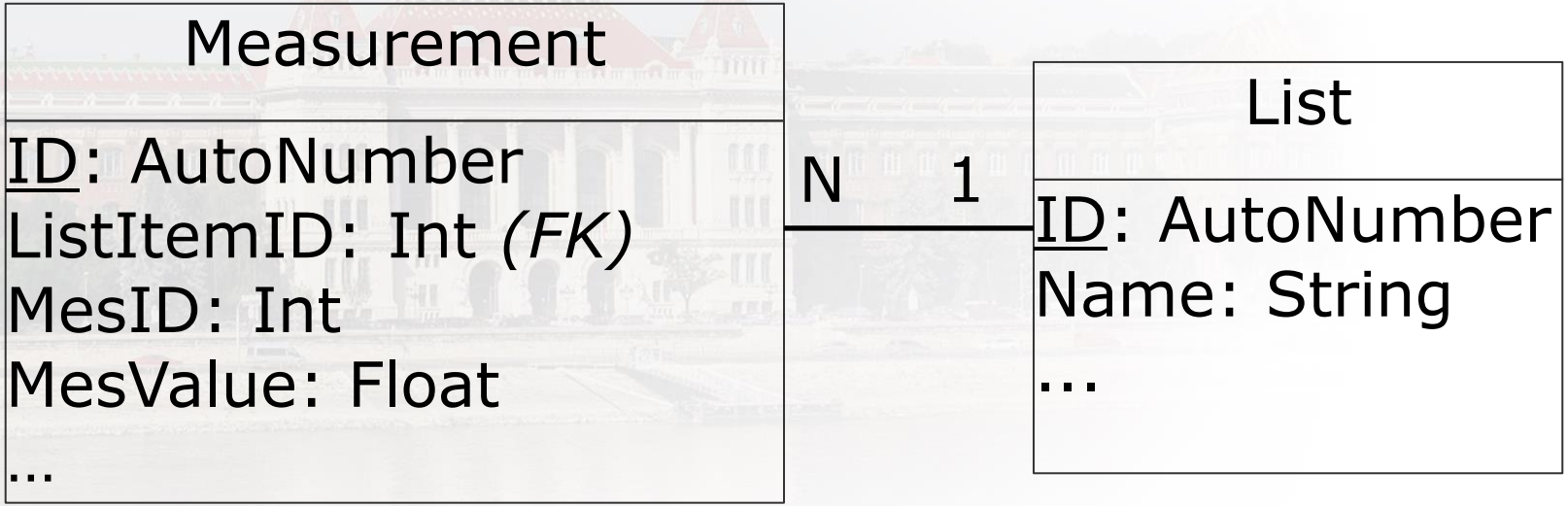
Measurement

ID	Sensor	Timestamp	Strain
1	Strain gauge1	2015.05.05 12:02:02	10
2	Strain gauge2	2015.05.05 12:02:02	10
3	Strain gauge1	2015.05.05 12:02:03	20
4	Strain gauge2	2015.05.05 12:02:03	30

Timestamps are regularly used to identify a measurement as an ID.

ID	Sensor	MesID	Strain
1	Strain gauge1	1	10
2	Strain gauge2	1	10
3	Strain gauge1	2	20
4	Strain gauge2	2	30

“Measurement” + “Report”



“Measurement” + “Report”

Measurement

ID	SensorID	Timestamp	Strain
1	1	2015.05.05 12:02:02	10
2	2	2015.05.05 12:02:02	10
3	1	2015.05.05 12:02:03	20
4	2	2015.05.05 12:02:03	30

Sensor

ID	Type	Others
1	Strain gauge1	...
2	Strain gauge2	...

"Measurement" – Pros&Cons

Pros:

— Easy to add a new sensor.

— Easier to analyze data obtained by different sensors.

— Less number of tables.

Cons:

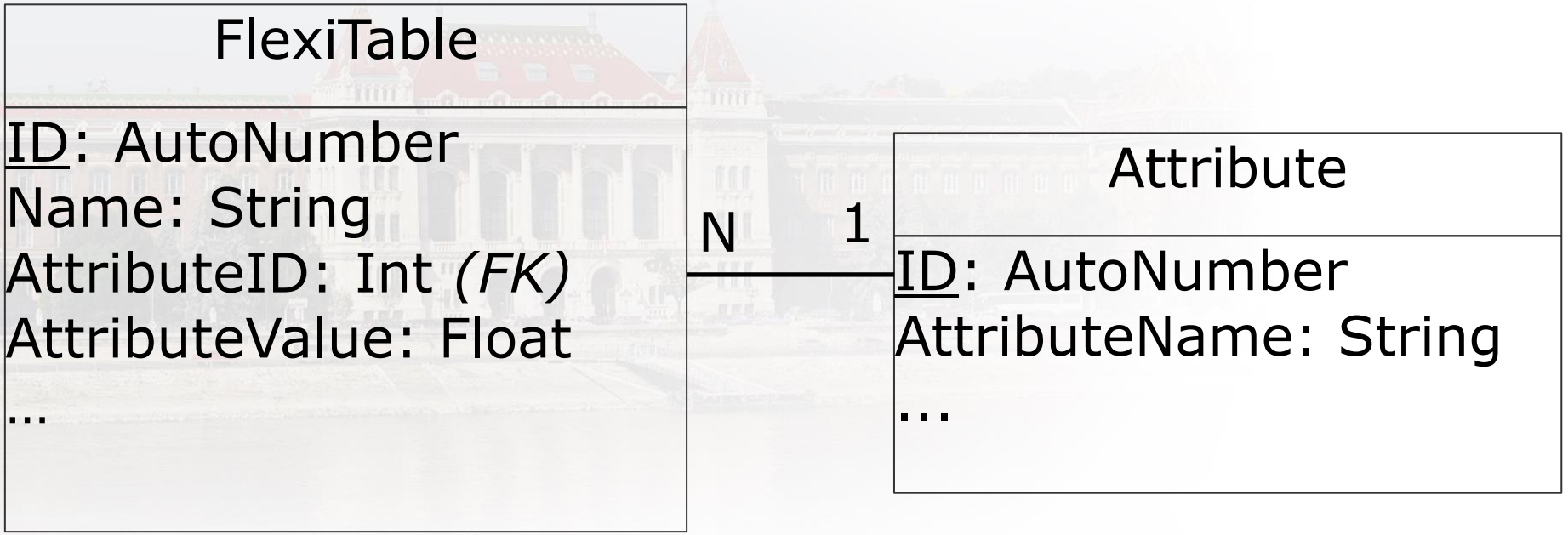
— To analyze a single sensor there is a need for filtering.

“Flexible table”

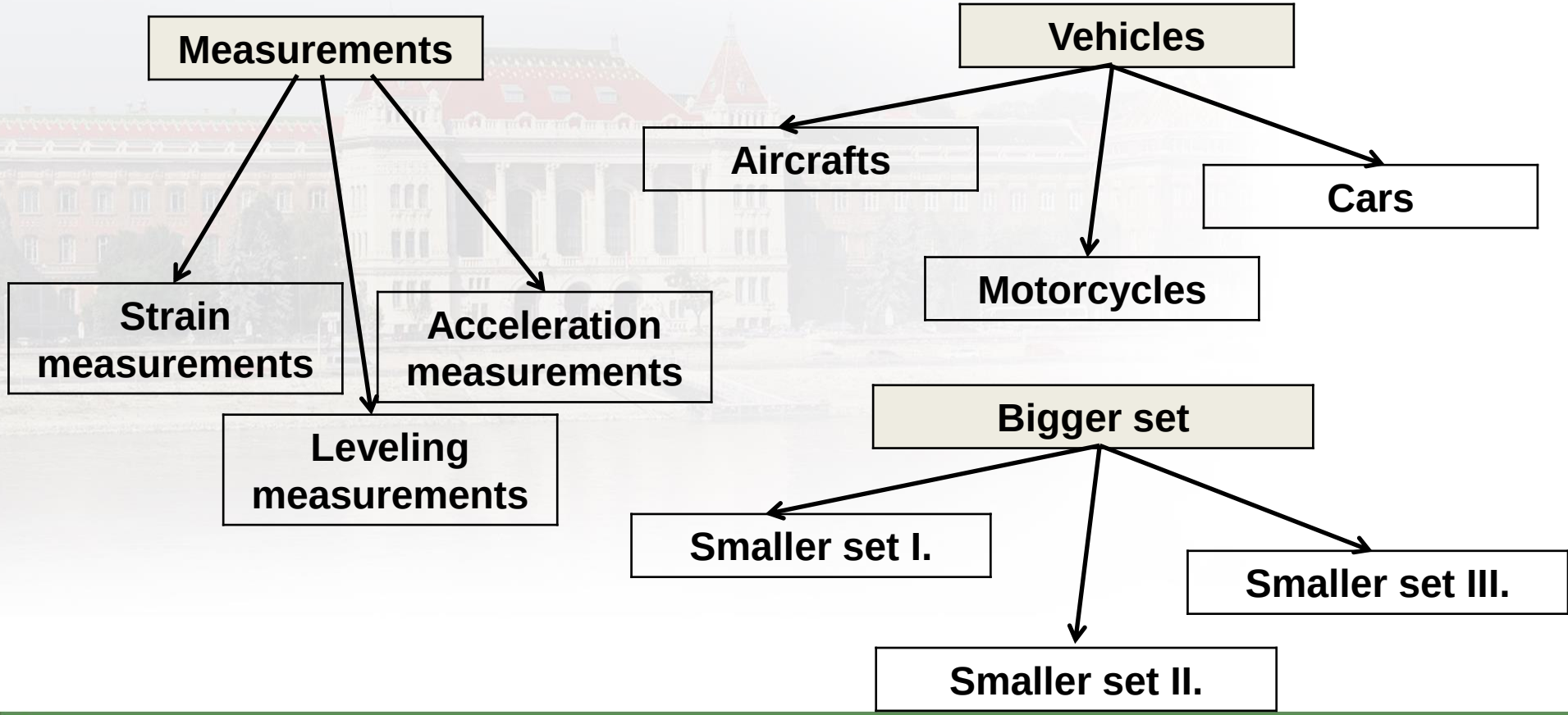
A common problem is that
- the attributes of a table may change later, such as
due to a change in specification,
- or a relation (table) has too many attributes,
- or NULL elements in the table often occur in different columns.

Solution: Add the attributes to a table. In the data table, enter the name of the attribute in one column and the corresponding values in another column.

“Flexible table”



STORING HIERARCHICAL DATA



“Flexible table”

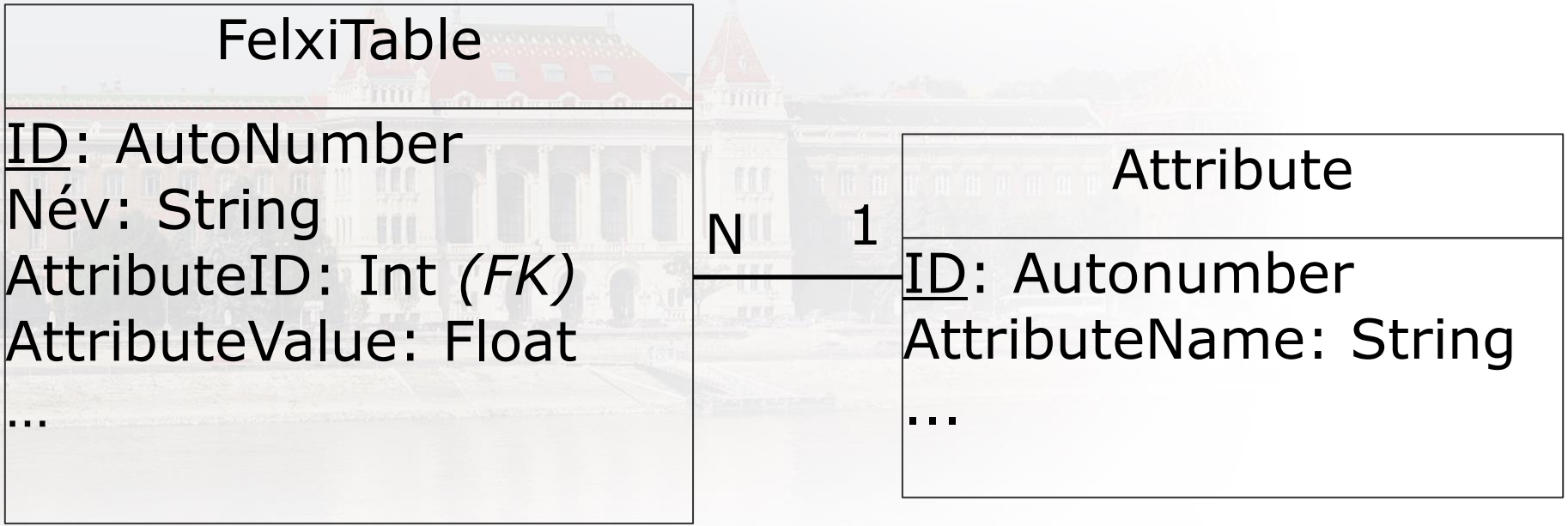
Measurement

ID	Sensor	Acceleration	Strain	Displacement
1	Strain gauge	NULL	10	NULL
2	Accelerometer	20	NULL	NULL
3	Leveler	NULL	NULL	120
4	Strain gauge	NULL	10	NULL

Vehicles

ID	Name	DoorCount	WingSpan	WheelCount
1	Car1	4	NULL	4
2	Motorcycle1	NULL	NULL	2
3	Car2	2	NULL	4
4	Aircraft1	8	120	3

“Flexible table”



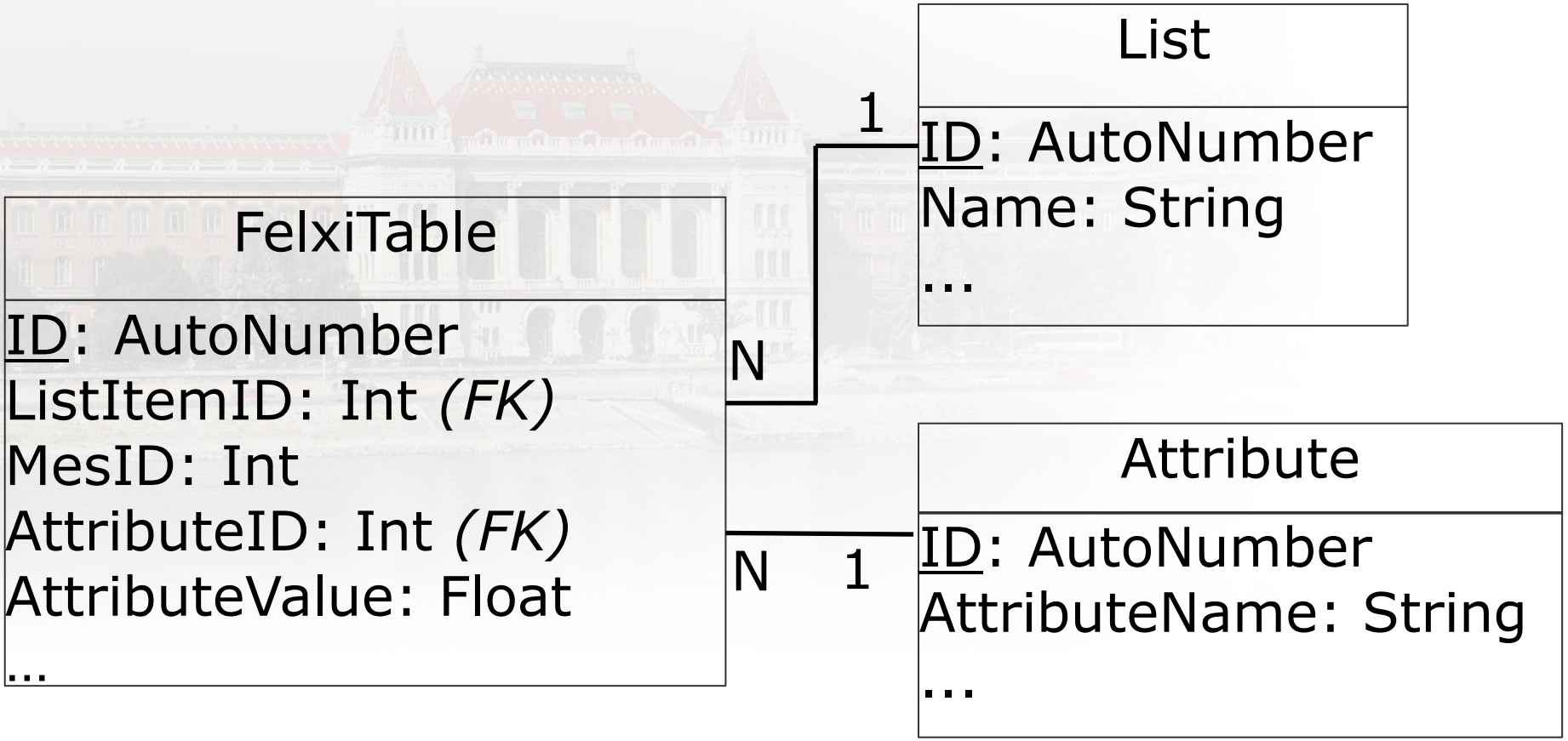
„Flexible table”+ „measurement”

ID	Sensor	MesTypeID	Value
1	Strain gauge	2	10
2	Accelerometer	1	20
3	Leveler	3	120
4	Strain gauge	2	10

Measurement Type

ID	MeasurementType
1	Acceleration
2	Strain
3	Displacement

„Flexible table” + „Measurement”+ „Report”



„Flexible table” + „Measurement”+ „Report”

ID	SensorID	MesID	MessTypeID	Value
1	1	1	2	10
2	2	1	1	20
3	3	1	3	120
4	1	2	2	10
5	3	2	3	35
6	1	3	2	125

Sensor

ID	SensorType
1	Strain gauge
2	Accelerometer
3	Leveler

MeasType

ID	MeasurementType
1	Acceleration
2	Strain
3	Displacement

„Flexible table” + „Measurement”+ „Report”

ID	Name	AttributeID	Value
1	Car1	1	4
2	Car1	3	4
3	Motorcycle1	3	2
4	Car2	1	2
5	Car2	3	3
6	Aircraft1	1	8
7	Aircraft1	2	120
8	Aircraft1	3	3

ID	Attribute
1	DoorCount
2	WingSpan
3	WheelCount

„Flexible Table” — Pros&Cons

Pros:

— Easy to add new attribute

— We can avoid to store plenty of NULL values

— We can handle hierarchical data together

Cons:

— Complex and difficult queries

— sometimes it is suboptimal

“Threshold”

Thresholds for measured values

There is no 1-1 match between values and thresholds, as the measured value rarely exactly matches the limit.

It is not possible to establish a connection in the relational database schema, so there is no connection between the two tables, there may be a “floating” table.

“Threshold” - data

Measurements | Sensors | Levels | Critical_gyro

measuementID	sensorID	time	value
1	1	2018. 10. 03.	20
2	2	2018. 10. 03.	27
3	2	2018. 10. 05.	30
4	1	2018. 10. 04.	25
5	2	2018. 10. 04.	23
6	3	2018. 10. 04.	29
7	3	2018. 10. 06.	21
8	1	2018. 10. 07.	19
9	2	2018. 10. 05.	27
10	2	2018. 10. 07.	28

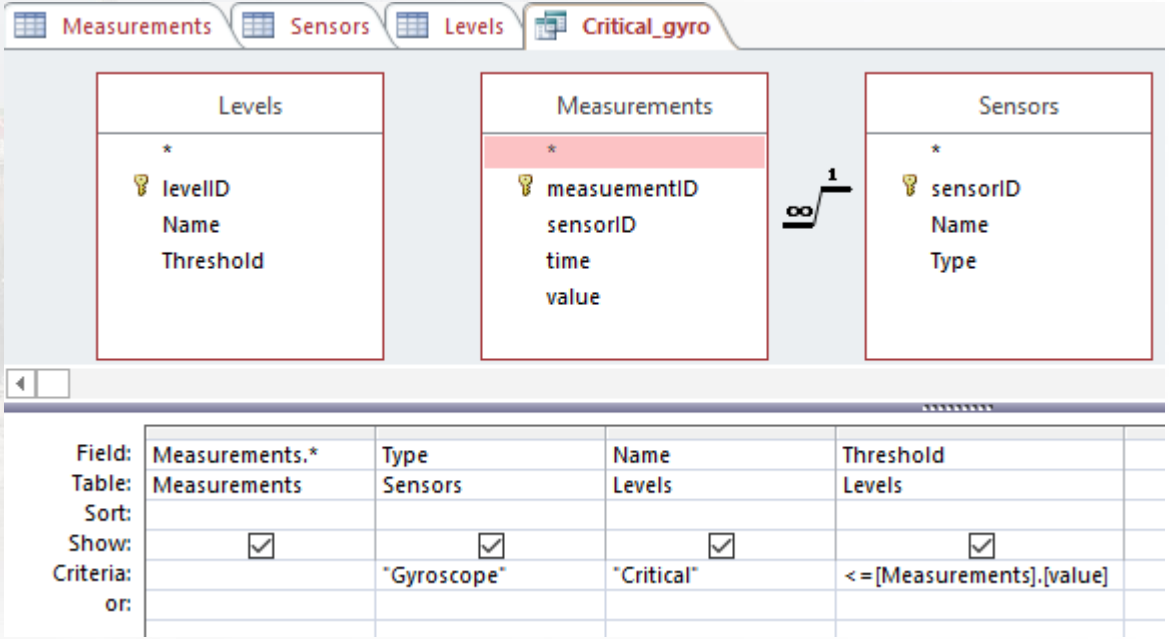
Measurements | Sensors | Levels | Critica

sensorID	Name	Type
1	Acc1	Accelerometer
2	Gyro1	Gyroscope
3	Gyro2	Gyroscope

Measurements | Sensors | Levels | Cri

levelID	Name	Threshold
1	Critical	28
2	Warning	26

“Threshold” – Relational schema diagram



Measurements | Sensors | Levels | Critical_gyro

measuementID	sensorID	time	value	Type	Name	Threshold
3	2	2018. 10. 05.	30	Gyroscope	Critical	28
6	3	2018. 10. 04.	29	Gyroscope	Critical	28
10	2	2018. 10. 07.	28	Gyroscope	Critical	28

Flexible table – in Access

BUDAPESTI MŰSZAKI
ÉS GAZDASÁGTUDOMÁNYI EGYETEM
Építőmérnöki Kar - építőmérnöki képzés 1782 óta

Fotogrammetria és Térinformatika Tanszék

Recovering by query

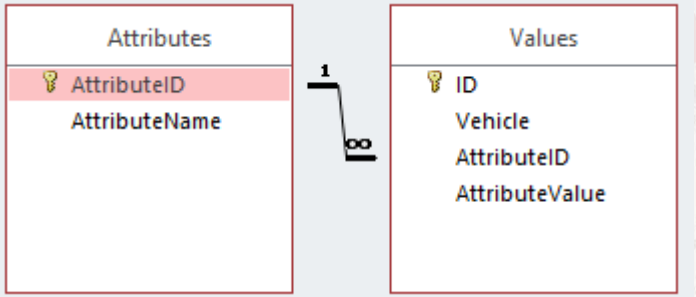
We stored all information, no data loss.

Easy to extend the structure with a new vehicle, vehicle type and attribute.

Hardly followable.

Result of a query does not require storage space!!!

What do we have?



Attributes

AttributeID	AttributeNa
1	DoorCount
2	WingSpan
3	WheelCount

Values

ID	Vehicle	AttributeID	AttributeVal
1	Car1	1	4
2	Car1	3	4
3	Motorcycle1	3	2
4	Car2	1	2
5	Car2	3	3
6	Aircraft1	1	8
7	Aircraft1	2	120
8	Aircraft1	3	3
(New)		0	0

We define three new set by separate queries

Vehicle	DoorCount
Aircraft1	8
Car1	4
Car2	2

AttributeID=1

Vehicle	WingSpan
Aircraft1	120

AttributeID=2

Vehicle	WheelCount
Aircraft1	3
Car1	4
Car2	3
Motorcycle	2

AttributeID=3

Ensure to have all vehicles – use extra query

Vehicle

Vehicle
Aircraft1
Car1
Car2
Motorcycle

C1

Vehicle	DoorCount
Aircraft1	
Car1	
Car2	

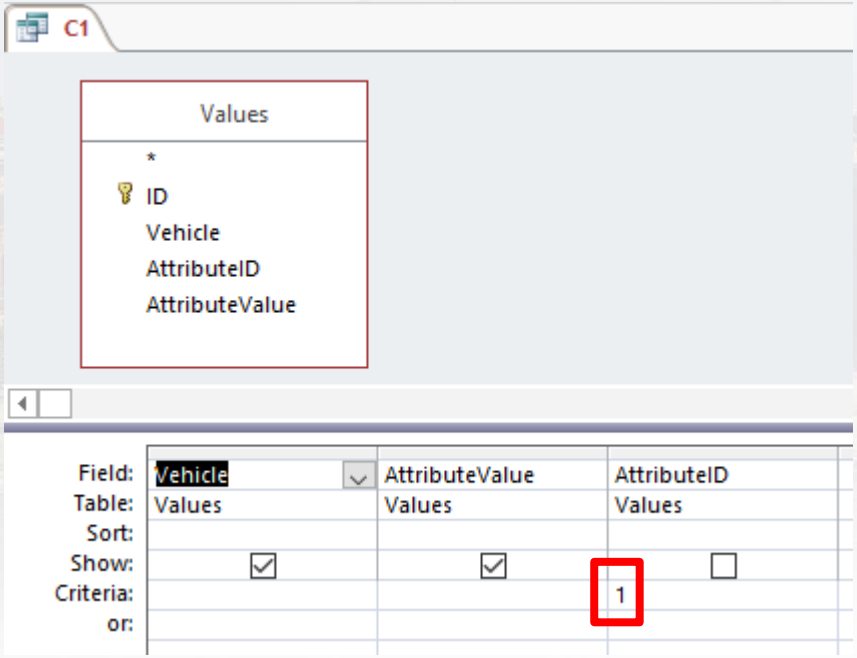
C1

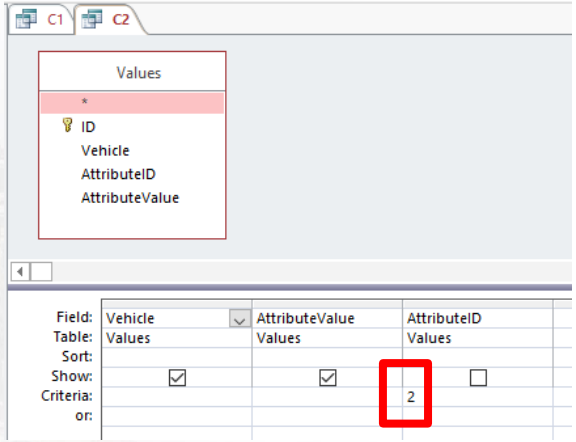
Vehicle	WingSpan
Aircraft1	120

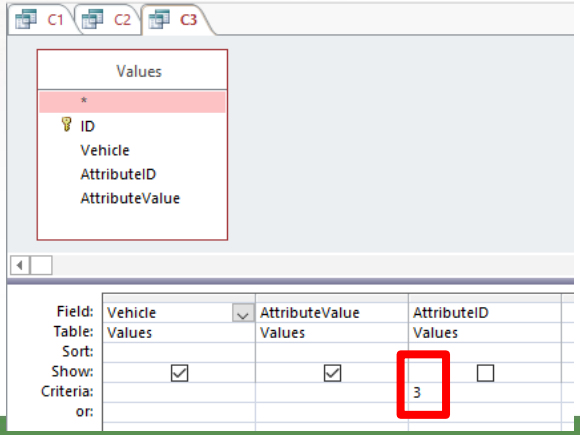
C3

Vehicle	WheelCount
Aircraft1	3
Car1	4
Car2	3
Motorcycle	2

Create designated columns

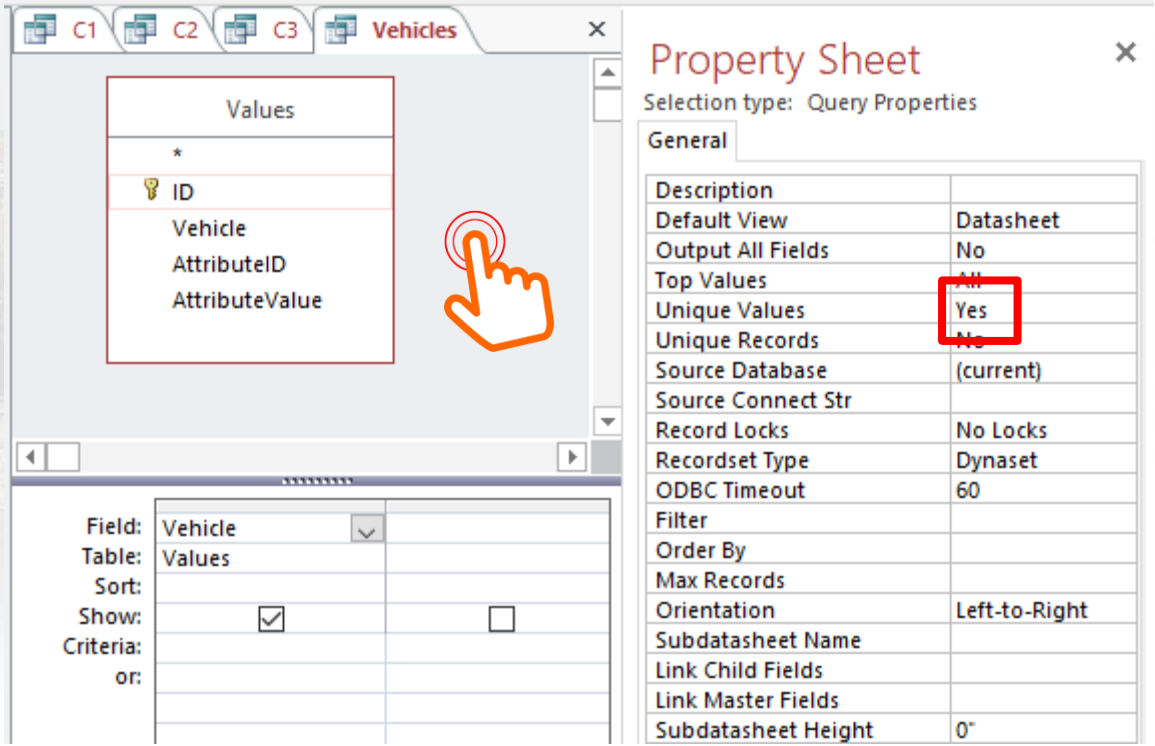




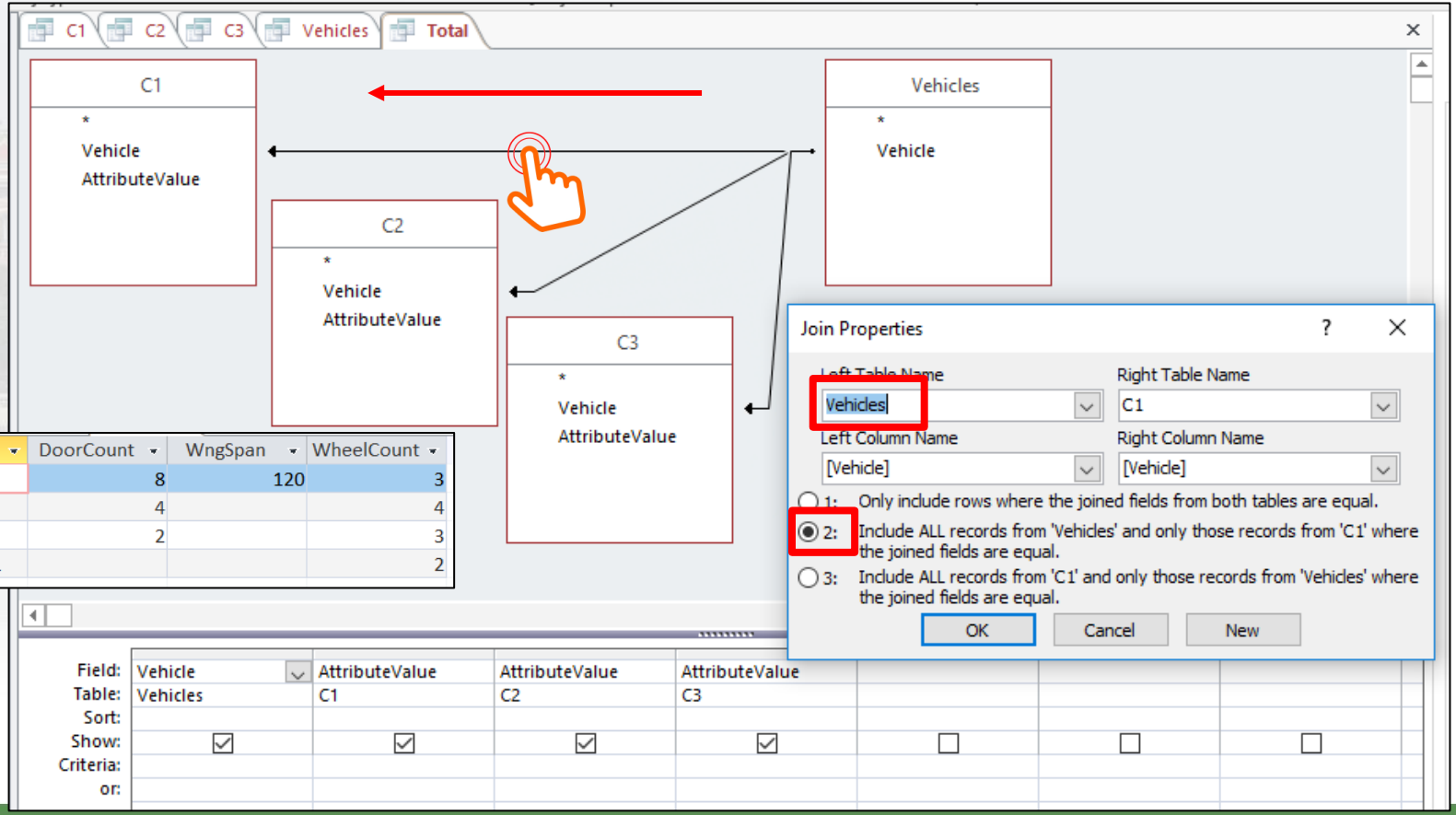


Missing attribute - value combinations are missing

List all vehicles



JOIN SUBQUERIES



Vehicle	DoorCount	WngSpan	WheelCount
Aircraft1	8	120	3
Car1	4		4
Car2	2		3
Motorcycle1			2

CONCLUSION

- Design patterns
- Practice: flexible table

Thank you for your attention!

Questions?

BUDAPESTI MŰSZAKI
ÉS GAZDASÁGTUDOMÁNYI EGYETEM
Építőmérnöki Kar - építőmérnöki képzés 1782 óta

Fotogrammetria és Térinformatika Tanszék