PITFALLS IN RELATIONAL DATABASES
connection trap (E.F. Codd 1970)

SP:

S#	P#
S1	P1
S2	P1
S2	P2

PJ:

P#	J#
P1	J1
P1	J2
P2	J3

SPJ:

S#	P#	J#
S1	P1	J1
S1	P1	J2
S2	P1	J1
S2	P1	J2
S2	P2	J3

JOIN CAN RESULT IN MEANINGLESS RESULTS

SPJ = SP [P# = P#] PJ

JOIN ON COMMON ATTRIBUTE P# IS MEANINGLESS

(S2 COULD BE THE ONLY SUPPLIER FOR PROJECT J1)

THIS IS CAUSED BY THE JOIN ON PARTIAL KEYS !!

RELATIONAL EXERCISES

The model concerns the registration of patient treatments in a hospital. The following relations are available:

***relation* patient**
(pat#, name, address, town)
***relation* physician**
(phys#, name, extension, dept#)
***relation* department**
(dept#, internal_address, extension)
***relation* treatment type**
(ttype, description, hourly_rate)
***relation* treatment**
(tm#, pat#, phys#, ttype, date, minutes_duration)
***relation* admission**
(adm#, pat#, phys#, admission_date, release_date).

SQL QUERY:
Determine the total treatment time per patient.

```
SELECT P . pat#, name, town, SUM (minutes_duration)
FROM treatment T, patient P
WHERE T . pat# = P . pat#
GROUP BY P . pat#, name, town
```

RELATIONAL EXERCISES

The model to be used in the exercise concerns the registration of patient treatments in a hospital. The following relations are available:

***relation* patient**
(pat#, name, address, town)
***relation* physician**
(phys#, name, extension, dept#)
***relation* department**
(dept#, internal_address, extension)
***relation* treatment type**
(ttype, description, hourly_rate)
***relation* treatment**
(tm#, pat#, phys#, ttype, date, minutes_duration)
***relation* admission**
(adm#, pat#, phys#, admission_date, release_date).

EXERCISE 1

Does a patient have to be registered for admission? Explain your answer briefly.

EXERCISE 2

Can a patient only have treatments by physicians belonging to one department? Explain your answer briefly.

EXERCISE 3

Formulate the following query by using SQL: Determine total treatment costs per patient per admission.

SQL EXERCISE
incorrect solution

relation **treatment type**

(ttype, description, hourly_rate)

relation **treatment**

(tm#, pat#, phys#, ttype, date, minutes_duration)

relation **admission**

(adm#, pat#, phys#, admission_date, release_date).

EXERCISE 3

Formulate the following query by using SQL:

Determine total treatment costs per patient per admission.

```
SELECT  A.pat#, admission_date,
        SUM (minutes_duration * hourly_rate / 60)
FROM    treatment type TT, treatment T, admission A
WHERE   A.pat# = T.pat#
        AND T.ttype = TT.ttype
        AND T.date >= A.admission_date
        AND T.date <= A.release_date
GROUP BY A.pat#, A.admission_date
```

SQL EXERCISE

treatment

tm#	pat#	phys#	ttype	date	min_dur	
1	1	12	A	2/4 ’92	15	
2	2	13	D	2/4 ’92	20	
3	2	14	E	4/4 ’92	10	(1)
4	2	15	A	5/4 ’92	15	
5	1	16	G	5/4 ’92	20	(2)

admission

adm#	pat#	phys#	adm_date	rel_date
1	1	12	2/4 ’92	4/4 ’92
2	2	14	1/4 ’92	4/4 ’92
3	2	15	4/4 ’92	8/4 ’92

PROBLEMS:

(1) CORRESPONDS WITH ADMISSION 2 OR 3 ?

(2) OUTSIDE ADMISSION PERIOD FOR THIS PATIENT (REFERENTIAL INTEGRITY IS NOT ENOUGH)

PITFALLS IN RELATIONAL DATABASES
empty sets (J.F. Sowa 1984 and W. Kent 1978)

EMPTY SETS LEAD TO UNACCEPTABLE CONCLUSIONS.

Example: EVERY UNICORN IS A COW:

$$\forall x \, (\text{unicorn}(x) \Rightarrow \text{cow}(x))$$

this is equivalent with:

$$\neg\exists x \, (\text{unicorn}(x) \wedge \neg \, \text{cow}(x))$$

CONSEQUENCE FOR DATABASES:

EVERY EMPTY RELATION, IS EQUAL TO ANY OTHER EMPTY RELATION

(THEY MAY CONTAIN DIFFERENT ATTRIBUTES) !!

SQL EXERCISE

items

ITEM#	DESCRIPTION	STOCK
1	CHAIR	20
2	TABLE	50
3	BOOKCASE	15

sales

ITEM#	WEEK	DAY	QTY
1	1	1	2
2	1	2	1
3	1	2	3
1	1	2	1
2	1	4	1
3	1	3	1

SQL QUERY:

Select items where the total sold quantity is more than the item's stock.

```
SELECT I . ITEM#
FROM SALES S , ITEMS I
WHERE S . ITEM# = I . ITEM#
GROUP BY I . ITEM#
HAVING SUM (S . QTY) > I . STOCK
```

SQL EXERCISE

items

ITEM#	DESCRIPTION	STOCK
1	CHAIR	20
2	TABLE	50
3	BOOKCASE	15

sales

ITEM#	WEEK	DAY	QTY
1	1	1	2
2	1	2	1
3	1	2	3
1	1	2	1
2	1	4	1
3	1	3	1
1	1	3	2
2	1	4	3
3	2	2	1
1	2	1	3
2	2	3	1
3	2	3	2
1	2	2	2
2	2	4	3
3	2	4	3
1	2	5	2
3	2	5	2

QUERY: Select items with descending sales figures (i.e. quantity in week 2 < quantity in week 1)

PITFALLS IN RELATIONAL DATABASES
SQL EXAMPLE

items

ITEM#	DESCRIPTION	STOCK
1	CHAIR	20
2	TABLE	50
3	BOOKCASE	15

sales

ITEM#	WEEK	DAY	QTY
1	1	3	10
2	1	4	5
3	1	5	5
1	2	2	5
3	2	4	10

QUERY:

Select items with descending sales figures.

```
SELECT I . ITEM#
FROM SALES S , ITEMS I
WHERE S . ITEM# = I . ITEM#
AND S . WEEK# = 2
GROUP BY I . ITEM#
HAVING SUM (S . QTY) <
      (SELECT SUM (S . QTY)
      FROM SALES S
      WHERE S . ITEM# = I . ITEM#
      AND S . WEEK# = 1)
```

RELATIONAL PITFALL (CONTINUED)
discussion

Select items with descending sales figures.

```
SELECT I . ITEM#
FROM SALES S , ITEMS I                 ||   ITEMS
WHERE S . ITEM# = I . ITEM#            ||   SOLD
AND S . WEEK# = 2                      ||   IN
GROUP BY I . ITEM#                     ||   WEEK 2
HAVING SUM (S . QTY) <
      (SELECT SUM (S . QTY)
      FROM SALES S
      WHERE S . ITEM# = I . ITEM#
      AND S . WEEK# = 1)
```

AS A CONSEQUENCE:

UNSOLD ITEMS (E.G. ITEM# 2, IN STOCK 50) ARE NOT FOUND.

MANAGEMENT DECISION BASED ON SQL:
UNMARKETABLE ITEMS REMAIN IN STOCK.

PITFALLS IN RELATIONAL DATABASES
compound keys

ROUTE (from_harbour, to_harbour, date, ship)

from_harbour	to_harbour	date	ship
AMSTERDAM	MARSEILLE	2.9.92	ANNA
AMSTERDAM	LISBOA	3.9.92	LOT

CARGO (from_harbour, to_harbour, date, ship, container)

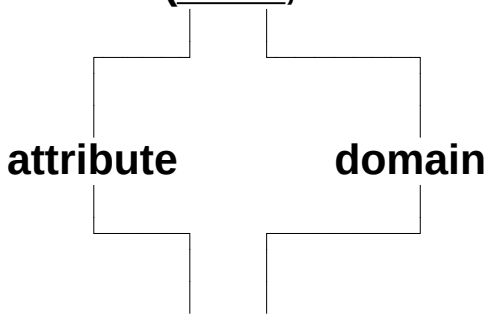
from_harbour	to_harbour	date	ship	container
AMSTERDAM	MARSEILLE	2.9.92	ANNA	12345678
AMSTERDAM	MARSEILLE	2.9.92	ANNA	87654321
AMSTERDAM	MARSEILLE	2.9.92	ANNA	43218765
AMSTERDAM	LISBOA	3.9.92	LOT	11223344

IT IS IMPOSSIBLE TO CHANGE THE DESTINATION IN TABLE ROUTE, BECAUSE THEN THE RELATIONSHIP WITH CARGO IS LOST.

PITFALLS IN RELATIONAL DATABASES
conceptual redundancy

RELATIONAL DBMSs REQUIRE DOMAIN DEFINITIONS FOR EACH ATTRIBUTE.

relation R (rx : x, ...)



relation S (sx : x, sy : y, ...)

THE FOLLOWING CONSTRAINTS HOLD:

1. DOMAIN CONSTRAINT S (sx) $\subset$ (x)

2. DOMAIN CONSTRAINT R (rx) $\subset$ (x)

3. REFERENTIAL CONSTRAINT S (sx) $\subset$ R (rx)

CONSTRAINT 1 IS REDUNDANT !

EXAMPLE OF INCREASING TIME / SPACE COMPLEXITY: JOIN OPERATION.

relation R (x, y)

relation S (y, z)

join R [y = y] S

O(n)

X	Y
X 1	Y 1
X 2	Y 1
X 3	Y 1

O(m)

Y	Z
Y 1	Z 1
Y 1	Z 2
Y 1	Z 3
Y 1	Z 4

O(n×m)

X	Y	Z
X 1	Y 1	Z 1
X 1	Y 1	Z 2
X 1	Y 1	Z 3
X 1	Y 1	Z 4
X 2	Y 1	Z 1
X 2	Y 1	Z 2
X 2	Y 1	Z 3
X 2	Y 1	Z 4
X 3	Y 1	Z 1
X 3	Y 1	Z 2
X 3	Y 1	Z 3
X 3	Y 1	Z 4

PITFALLS IN RELATIONAL DATABASES
performance (continued)

relation **participation**

(employee#, project#, function)

relation **activities**

(employee#, project#, date, activity type, hours)

QUERY: For each participation determine the total hours worked.

```
SELECT P.employee#, P.project#, P.function, SUM (hours)
FROM participation P, activities A
WHERE P . employee# = A . employee#
  AND P . project# = A . project#
GROUP BY P . employee#, P . project#, P . function
```

2 SEPARATE JOINS

CARDINALITIES FOR 200 WORKING DAYS:

employees	**100**	
projects	**50**	
participations	**500 :**	**5 per employee, 10 per project**
activities	**40 000 :**	**400 per employee, 800 per project**
intermediate	**200 000 - 400 000**	**(400 - 800 * final result)**
final result	**500**	

PITFALLS IN RELATIONAL DATABASES
lack of orthogonality (C.J. Date 1986)

EXAMPLE:

SUPPLIER S (S#, SNAME, ...)

PART P (P#, PNAME, ...)

SUPPLY SP (S#, P#, ...)

GET SUPPLIERS WHO SUPPLY PART P2.

1.
```
SELECT  SNAME
FROM    S
WHERE   S# IN
        (SELECT     S#
        FROM    SP
        WHERE   P# = P2)
```

2.
```
SELECT  SNAME
FROM    S
WHERE   S# = ANY
        (SELECT     S#
        FROM    SP
        WHERE   P# = P2)
```

3.
```
SELECT  SNAME
FROM    S
WHERE   EXISTS
        (SELECT     *
        FROM    SP
        WHERE   S# = S.S# AND P# = P2)
```

PITFALLS IN RELATIONAL DATABASES
lack of orthogonality (continued)

```
4.  SELECT  DISTINCT SNAME
    FROM    S, SP
    WHERE   S.S# = SP.S# AND P# = P2

5.  SELECT  SNAME
    FROM    S
    WHERE   0 <
            (SELECT     COUNT (*)
            FROM    SP
            WHERE   S# = S.S# AND P# = P2)

6.  SELECT  SNAME
    FROM    S
    WHERE   P2 IN
            (SELECT     P#
            FROM    SP
            WHERE   S# = S.S#)

7.  SELECT  SNAME
    FROM    S
    WHERE   P2 = ANY
            (SELECT     P#
            FROM    SP
            WHERE   S# = S.S#)
```

THIS DESIGN INSTABILITY MAKES IT DIFFICULT TO COOPERATE IN A COMMON PROJECT.

PITFALLS IN RELATIONAL DATABASES
lack of orthogonality (continued)

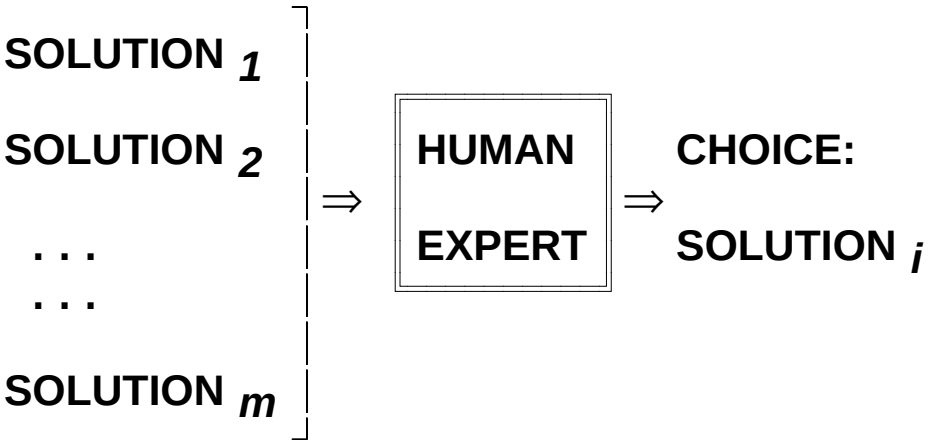
SITUATION:

ONE PROBLEM WITH SEVERAL SOLUTIONS.

RESULT:

DIFFERENT PERFORMANCE CHARACTERISTICS

IN GENERAL:



CHOICE IS NOT ONLY DBMS DEPENDENT BUT ALSO DBMS VERSION DEPENDENT.

REQUIRES AN EXPERT SYSTEM (WITH KNOWLEDGE OF DBMS OPTIMIZERS) IN CASE QUERIES ARE GENERATED BY A COMPUTER PROGRAM.

PITFALLS IN RELATIONAL DATABASES
meta data

PRIMARY KEY IS A PROPERTY OF EACH RELATION, HENCE THE DATA DICTIONARY SHOULD CONTAIN AT LEAST THE FOLLOWING RELATION:

relation (relation name, primary key, ...)

EXAMPLE DATABASE:

SUPPLIER (S#, SNAME, ...)
PART (P#, PNAME, ...)
SUPPLY (S#, P#, ...)

THE DATA DICTIONARY SHOULD CONTAIN AT LEAST:

relation name	primary key	.....
SUPPLIER	S#	
PART	P#	
SUPPLY	(S#, P#)	

THIS RELATION IS NOT NORMALIZED !!

PITFALLS IN RELATIONAL DATABASES
inherent integrity

RELATIONAL ALGEBRA IS BASED ON MATHEMATICAL STRUCTURES, AND NOT ON STRUCTURES AS FOUND IN DATABASES.
(INCLUDING REFERENTIAL CONSTRAINTS).

HOWEVER, IT IS IMPOSSIBLE TO DERIVE CERTAIN INFORMATION WITHOUT AN ASSUMPTION OF THESE CONSTRAINTS.

EXAMPLE:

RELATIONS P, Q, R WITH ATTRIBUTES X, Y, ...

P (X, ...)
Q (X, Y, ...)
R (X, ...)

QUERY: GET Ys CORRESPONDING WITH ALL Xs.

Q [X ÷ X] P

THIS SOLUTION IS VALID IF AND ONLY IF REFERENTIAL INTEGRITY HOLDS.
(IT IS ASSUMED THAT ALL Xs ARE IN P).

PITFALLS IN RELATIONAL DATABASES
NULL values (C.J. Date 1990)

THE RELATIONAL MODEL DOES NOT CONTAIN EQUIVALENTS FOR GENERALIZATION AND SPECIALIZATION ABSTRACTIONS.

EXAMPLE:
SPECIALIZATION IS REQUIRED WHEN NOT ALL PROPERTIES ARE APPLICABLE TO ALL INDIVIDUALS.

THIS LEADS TO THE USE OF NULL VALUES:

NULL ≠ " " (BLANK),
NULL ≠ 0 (ZERO),
NULL ≠ any other value.

BUT NULL = NOT EXISTING.

HOWEVER:

THIS RESULTS NOT IN 3-VALUED LOGIC (true, false and unknown), BUT IN FACT IN ∞-VALUED LOGIC (with an infinite number of logical values).

PITFALLS IN RELATIONAL DATABASES
NULL values and SET FUNCTIONS

ANOMALIES APPEAR WHEN USING NULL VALUES

AND SET FUNCTIONS IN SQL.

ITEMS:

ITEM#	DESCRIPTION	QTY1	QTY2
111	CHAIR	10	30
222	TABLE	NULL	40
333	BOOKCASE	20	50

FIRST STEP:

SELECT SUM (QTY1) FROM ITEMS

RESULT: 10 + NULL + 20 = 30.

SECOND STEP:

SELECT SUM (QTY2) FROM ITEMS

RESULT: 30 + 40 + 50 = 120.

TOTAL:

SELECT SUM (QTY1 + QTY2) FROM ITEMS

RESULT: 40 + NULL + 70 = 110.

CONCLUSION:

SUM (QTY1) + SUM(QTY2) $\neq$ SUM (QTY1 + QTY2).

30 + 120 $\neq$ 110.

PITFALLS IN RELATIONAL DATABASES

- **CONNECTION TRAP**
  problem concerning the database structure.
- **EMPTY SET**
  sets are not suitable for describing a database.
- **IDENTIFIERS**
  normal attributes are not suitable for identification.
- **CONCEPTUAL REDUNDANCY**
  undesirable distinction between attribute and domain.
- **UNIVERSAL RELATION**
  normal forms are not allowed.
- **TIME/SPACE COMPLEXITY**
  performance inherent to relational operations.
- **SQL PITFALLS**
  language leads easily to misinterpretations.
- **ORTHOGONALITY**
  the language lacks orthogonality.
- **META DATA**
  relational databases are not self-describing.
- **INHERENT INTEGRITY**
  referential constraints should be inherent.
- **MISSING DATA**
  null results in huge theoretical/practical problems.
- **DESIGN INSTABILITY**
  caused by missing generalization/specialization.

EXERCISE

A trading company wishes to record the address of each of its suppliers in a database. The following relations were defined during the relational design:

***relation* SUPPLIER (sup#, name, premise#)**
***relation* PREMISE (premise#, address, town)**

where premise# in SUPPLIER refers to premise# in PREMISE. A snapshot of the database is shown below. The database contents comply with the integrity requirements of the relational model.

a Which integrity requirement could be added meaningfully to this model definition under the given circumstances?

b What consequences would this addition have for update commands?

c Provide another relational model satisfying the information requirements in a simpler manner.

SUPPLIER:

sup#	name	premise#
S01	Broadwick	P01
S02	Narroton	P02
S03	Bapchill	P03

PREMISE:

premise#	address	town
P01	12, High Street	Maretown
P02	7, Abbey Terrace	Ennisfray
P03	12, High Street	Maretown
P04	2, Broadway	Swindon