data design separated from process design

the effect of process structure omitted for data design

relationships between data and processes are of the first-order type
thus the final ‘technical’ design is achieved by linear superposition
benefits

- Data redundancy minimized
- Data shared amongst applications
- Data maintained centrally
- Common processes between applications
- Application software transparent

Relational Model

\[ R = (r_1, r_2, \ldots, r_{i-1}, r_i, r_{i+1}, \ldots, r_n) \]

- none of \( r_i = (r_{i1}, r_{i2}, \ldots) \)

- \( \forall k = (r_i, r_j, \ldots) : [\Pi R(k)] \equiv [R] \)

- \( \exists k = (r_i, r_j, \ldots) \)

- \( \{ R_i \} \) is closed under \( \Pi, \sigma, \eta, \ldots \)

- FD: \( X \rightarrow Y \) holds for \( R = (\ldots, X, Y, \ldots) \)

\[ \forall x \in X, [\Pi \sigma R(X=x) Y] \leq 1 \]
Relational Optimisation

\{R_i\} \rightarrow \{S_k\}

- No transaction may
  - cause loss of information delete anomaly
  - violate entity integrity insert anomaly
  - carry any risk of inconsistent updating update anomaly
- Minimised data redundancy

Boyce-Codd Normal Form

A 1NF relation \(R(X_1, X_2, ..., X_n)\) is in BCNF iff:
for every attribute collection \(X\) of \(R\)
if any attribute not in \(X\) is functionally dependent on \(X\)
then all attributes in \(R\) are functionally dependent on \(X\)
apply to a pair of union compatible relations

finite set-operations

complementary algebra operations

Relational Closure

Unary

Binary

Relational Algebra

UNION

DIFFERENCE

INTERSECTION

PRODUCT

PROJECT

RESTRICTION

SELECTION

JOIN

DIVISION

apply to a pair of relations with comparable attributes

PROJECT

SELECT

RESTRICT
JOIN
\( X = S \)

DIVIDE

[example] :-)
\texttt{print subject}
\begin{tabular}{l}
\texttt{class class_name} \\
C1 & FICTION \\
C2 & SCIENCE-FICTION \\
C3 & NON-FICTION \\
C4 & SCIENTIFIC \\
C5 & POETRY \\
C6 & DRAMA
\end{tabular}

Message: Relation subject returned.

[example] :-)
\texttt{print book}
\begin{tabular}{ll}
\texttt{reference author title} \\
R003 & JOYCE & ULYSSES \\
R004 & JOYCE & ULYSSES \\
R023 & GREENE & SHORT STORIES \\
R025 & ORWELL & ANIMAL FARM \\
R013 & LEM & ROBOTS TALES \\
R014 & LEM & RETURN FROM THE STARS \\
R016 & GOLDING & LORD OF THE FLIES \\
R028 & KING & STRENGTH TO LOVE \\
R143 & HEMINGWAY & DEATH IN THE AFTERNOON \\
R149 & HEMINGWAY & TO HAVE AND HAVE NOT
\end{tabular}


[example] :-)
\texttt{print index}
\begin{tabular}{lll}
\texttt{author title class shelf} \\
JOYCE & ULYSSES & C1 12 \\
GREENE & SHORT STORIES & C1 14 \\
ORWELL & ANIMAL FARM & C1 12 \\
LEM & ROBOTS TALES & C2 23 \\
LEM & RETURN FROM THE STARS & C2 23 \\
GOLDING & LORD OF THE FLIES & C1 12 \\
KING & STRENGTH TO LOVE & C3 24 \\
HEMINGWAY & DEATH IN THE AFTERNOON & C3 22 \\
HEMINGWAY & TO HAVE AND HAVE NOT & C1 12
\end{tabular}

Message: Relation index returned.
[example] :-)
b = (project (subject) (class))
difference
(project (index) (class))
Message: Relation b returned.
[example] :-)
print b

class
-----
C4
C5
C6
Message: Relation b returned.
[example] :-)
quit
Message: Closing [s] database.....

[example] :-)
a = project (select (join (subject)(index))
(subject.class = index.class)
(class_name = 'SCIENCE-FICTION'))
(title)
Message: Relation a returned.
[example] :-)
print a

title
------------------------
ROBOTS TALES
RETURN FROM STARS
Message: Relation zszxic returned.
[example] :-)
quit
Message: Closing [s] database.....
Why relational model has been so attractive?

- separation of physical & logical aspects
- data - process independence
- high level of data abstraction
- universal & uniform data structure
- global behavioural rules
- set of higher-level operations
- structure optimisation algorithm

How was it possible

- data - process independence
  - divorce from ADT
  - the only sensible way to do it:
    - make all operations universally applicable to every structure
    - have one universal primitive
Conclusion

any kind of ordering (set inclusion, tree, graph, convolution) imposed on a structure contradicts relational foundations

→ evolution of RDB imminent

Objectives

• structural simplicity → structural uniformity (regularity)
• separation of logical and physical aspects of database processing
• set-oriented processing → algebra-oriented processing
**M_0** = \{ADT, procedures\}

**M_1** = \{relations, r-operations\}  \hspace{1cm} E. Codd

**M_2** = \{nested relations, xr-operations\}  \hspace{1cm} H. Korth *et al.*

**M_3** = \{L, operations\}  \hspace{1cm} D. Scott

**M_4** = \{algebra (components, operations), transformations\}

---

**Abrial’s Binary Model**

**KNOWLEDGE :=**

**ELEMENTARY FACTS**
- John Doe was born in London on 19 Nov 1962
- The car with a number plate B1 BYE is a Ferrari

**SIMPLE RULES**
- Every man has necessarily two parents of whom he is the child
- A person has sometimes a spouse and if X is the spouse of Y then Y is the spouse of X
- A car has (if any) only one owner. Conversely, an owner may have zero, one or several cars

**COMPLEX RULES**
- The sex of a person is not subject to any change
- A single person who marries may not be single again in the future
- A person may not be, at a given time, in two different places

**DEDUCTIVE RULES**
- if x > y then BIG = x else BIG = y
- square() = twice (twice ())
WHEN THE MODEL DOES NOT KNOW A FACT OR A LAW ABOUT REALITY
THIS DOES NOT MEAN THAT THIS FACT OR LAW DOES NOT EXISTS.

CONSEQUENCE:

IF

THE MODEL HAS EXACTLY THE SAME KNOWLEDGE OF TWO OBJECTS
IT DOES NOT FOLLOW THEY ARE ONE AND THE SAME OBJECT.

THEREFORE

AN OBJECT ENTERING THE ‘PERCEPTION FIELD’ OF THE MODEL MUST
IDENTIFY ITSELF AS either NEW OBJECT or ALREADY KNOWN OBJECT.

THE DESCRIPTION OF AN OBJECT INSIDE THE MODEL IS GIVEN VIA THE
CONNECTIONS (access functions) IT HAS WITH OTHER OBJECTS.

person_of_sex (MALE) = (JOHN, PETER)
person_of_sex (FEMALE) = (JANE, MARY)
age (JOHN) = (27)

person_of_age (50) = (PETER, MARY)
child (PETER) = (JANE)

parent (JANE) = (PETER, MARY)

...
CATEGORIES

JOHN, JANE, PETER, MARY are PERSONs
27, 50, 20 are NUMBERs
MALE, FEMALE are SEXes

THUS, THE STRUCTURE OF THE EXAMPLE CAN BE ABSTRACTED INTO:

AND FURTHER STILL INTO:

CONNECTIONS MAY THEMSELVES REQUIRE SOME INFORMATION

EXAMPLE: PETER was_invited_by (PAUL and JANE) to PARIS on 15Jul1993

THIS CAN BE DESCRIBED BY BUILDING A NEW CATEGORY: INVITATION
AND THE FOLLOWING STRUCTURE:
defn CATEGORIES

PERSON = cat  there is new category
JOHN = generate PERSON  create new object of category
x ← generate PERSON
kill JOHN, kill x

NUMBER
PERSON
SEX

spous e
child
parent
spous e
age
sex
of-sex
of-age

r1 = rel (PERSON, SEX, sex = fun(1, 1), of_sex = fun(0, ∞))
r2 = rel (PERSON, NUMBER, age = fun(1, 1), of_age = fun(0, ∞))
r3 = rel (PERSON, PERSON, spouse = fun(0, 1), spouse)
r4 = rel (PERSON, PERSON, parent = fun(2, 2), child = fun(0, ∞))

a person has exactly one sex, one age, two parents, zero or one spouse and any number of children