



2E to Plex and Onwards

A Journey in Abstraction

Simon Williams

Founder, Synon Corporation



Where It Began

March Computer Systems

- Founded in 1975
- Fixed price development of custom applications
for IBM System/32, 34, 38 & 36
- 26 unique implementations in eight years
Fashion, pharmaceuticals, jewellery, cosmetics, shipping,
Legal, music, airline, chemicals, banking, food, publishing
- Developed productivity tools out of necessity



Philosophy

- Every enterprise can benefit from applications tailored to its precise needs
- The high cost of development forces too many enterprises to adopt the wrong solutions
- Application development must become ever more rapid and efficient



Synon – 1984 to 1998

- Founded in 1984 to develop Synon/2
- Launched Synon/1 in 1985; Synon/2 in 1986
- From zero in 1984 to \$34 million in 1989
- Acquired by Sterling in 1998: \$80m revenue
- Sterling was acquired by CA in 2000



Synon/2 (now CA 2E)

Synon/2's design principles

- IBM System/38
 - Built-in RDBMS, Object model, Future-proof architecture
- Data-driven approach to design
 - Users understand the structure of their data
- Productivity through abstraction
 - A higher level of abstraction than 3GLs
- Object orientation
 - Database access should be encapsulated for integrity

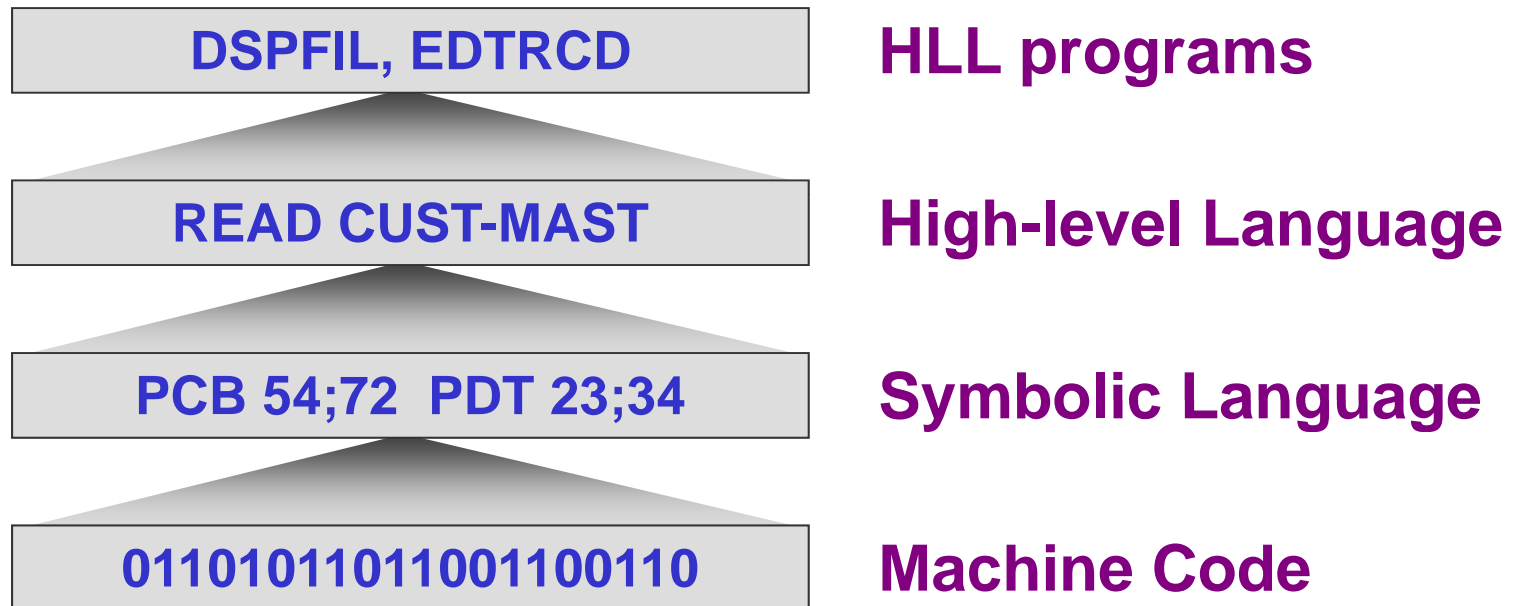


Object Orientation in 2E

- Encapsulation – no direct database access
 - Only via CRTOBJ; CHGOBJ; DLTOBJ
- Inherited behaviour through reusable functions
 - EDTRCD, EDTFIL, EDTTRN, DSPRCD, DSPFIL etc



Abstraction in 2E



Bigger building blocks = Greater productivity



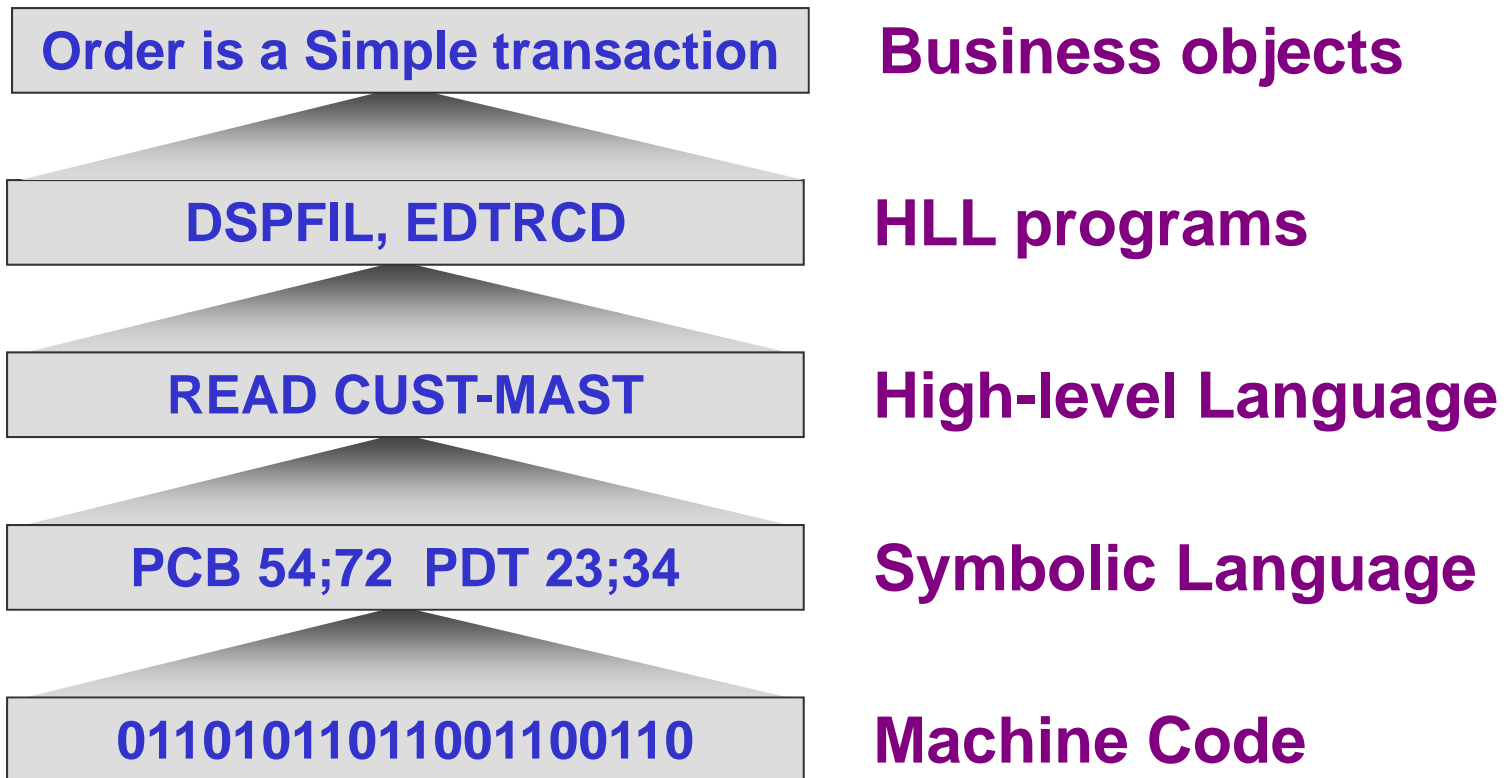
Obsydian (now CA Plex)

Obsydian's design principles

- Client/server platform
 - Aiming for implementation independence
- Preserving the best of Synon/2E
 - Fundamental principles had been proved sound
- Abstract business objects
 - The next level of abstraction for more productivity
- User-defined abstractions
 - Greater power & flexibility



Abstraction in Plex



Bigger building blocks = Greater productivity

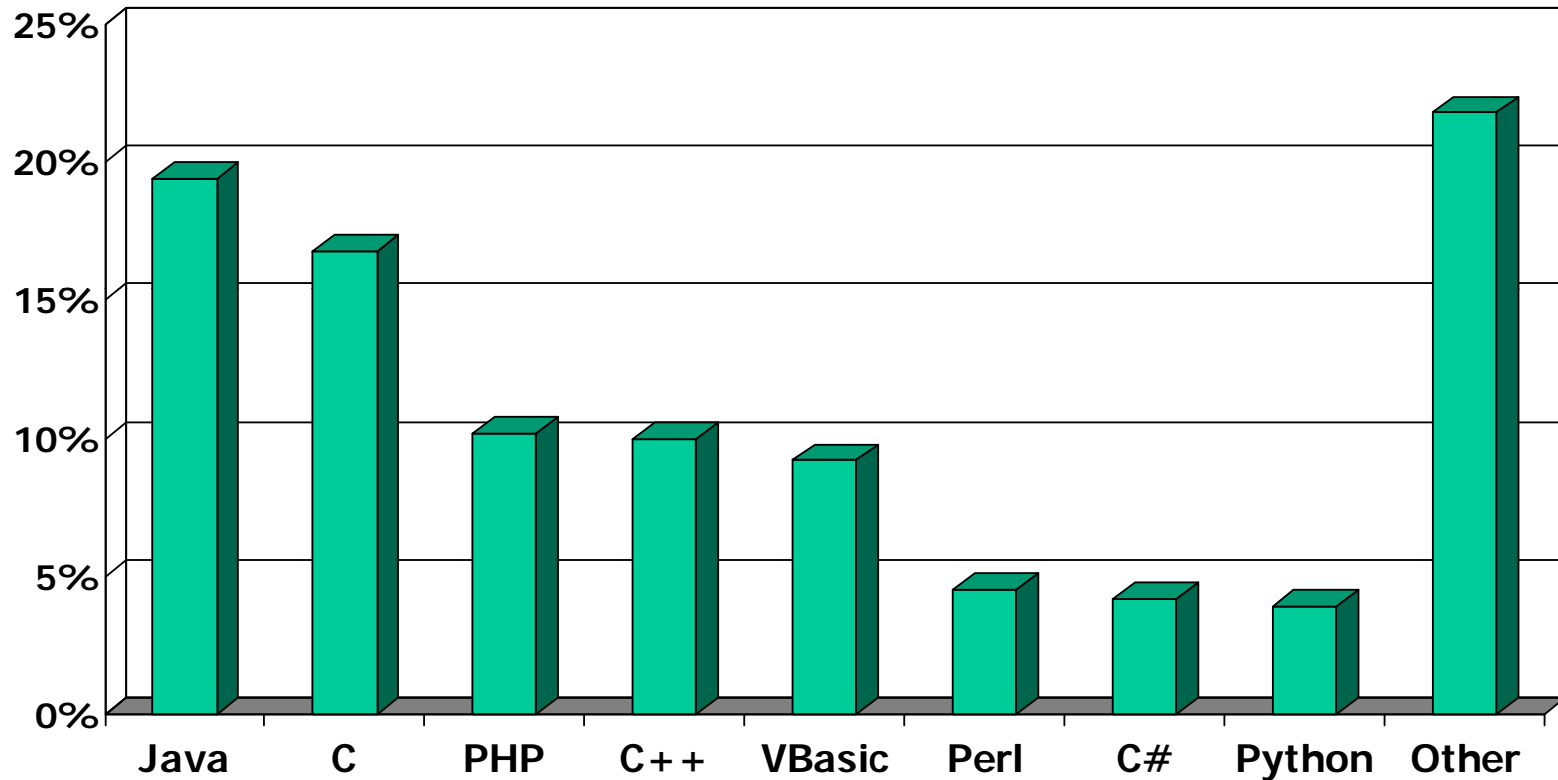


Thoughts on 4GLs

- Emerged in the '70s and '80s
 - On mainframe and mid-range computers
- Not true programming languages
 - Lack Turing-completeness
 - Aimed for greater productivity in a specific domain
- Developed on stable platforms
- Fell out of favour in the 1990s
 - Rapidly changing technology landscape
 - Client/server, UNIX, Internet, SOA, SaaS, Web 2.0



Languages Today



TIOBE Programming Community Index: number of skilled engineers, courses and third party vendors



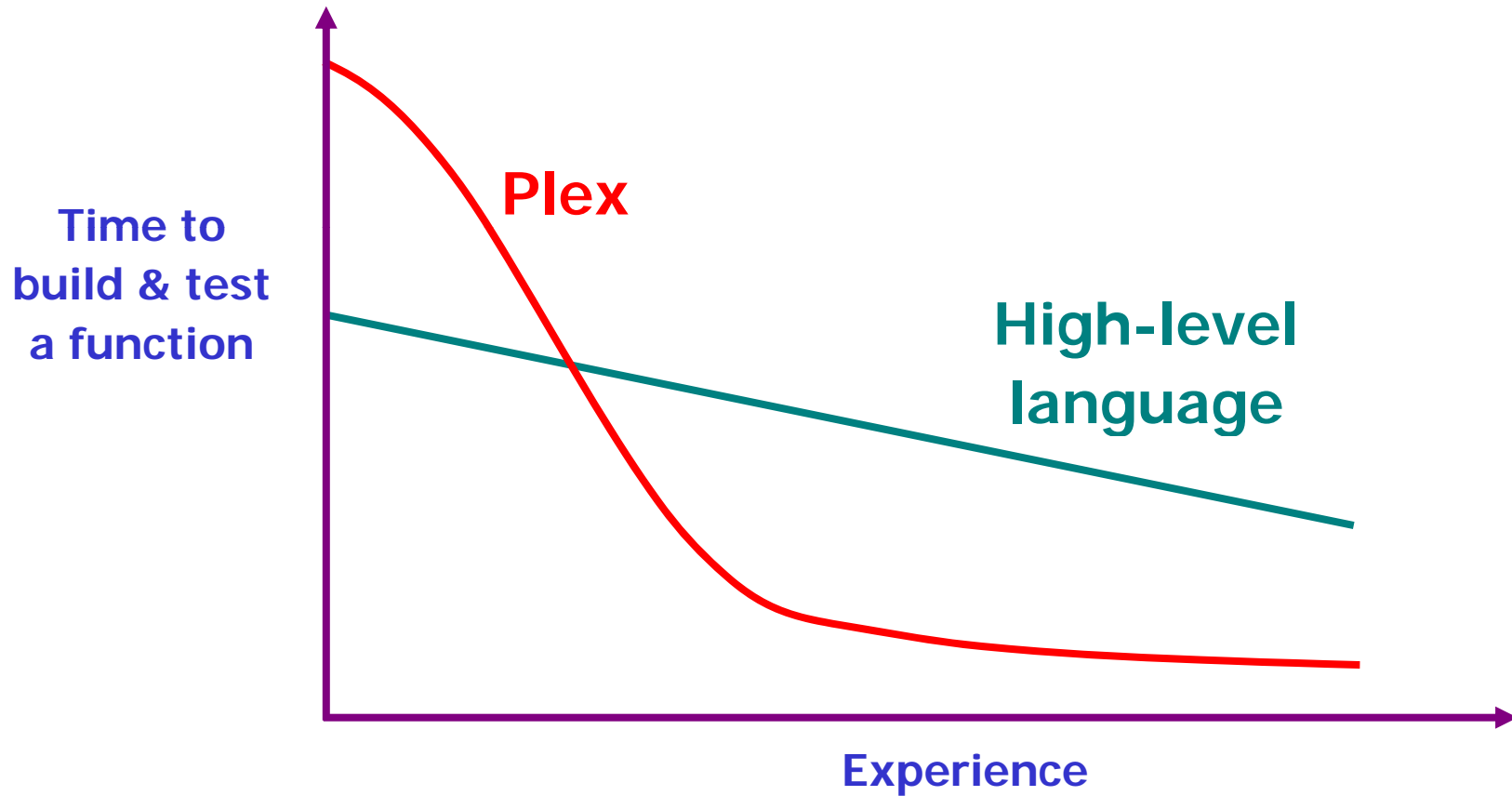
...and don't forget COBOL

- Processes 75% of the world's business data
...and 90% of global financial transactions
- 200 billion lines of code in live operation
- 5 billion new lines added every year
- More than 1.5 million developers globally

Source: Datamonitor, November 2008

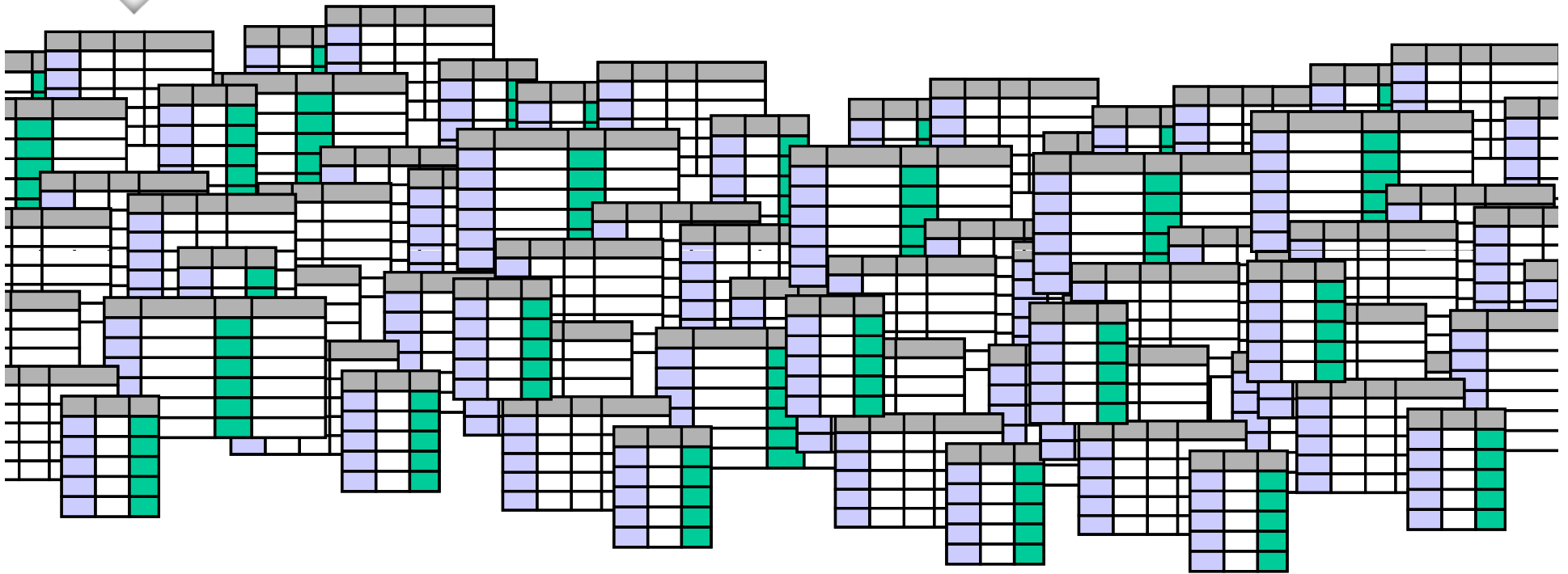


Complexity in Plex





Complexity in Applications



More than 16,500 tables in SAP



Simplicity

“Everything should be made as simple as possible, but not one bit simpler”

Albert Einstein

“Simplicity is complexity resolved”

Constantin Brancusi



Beyond Plex

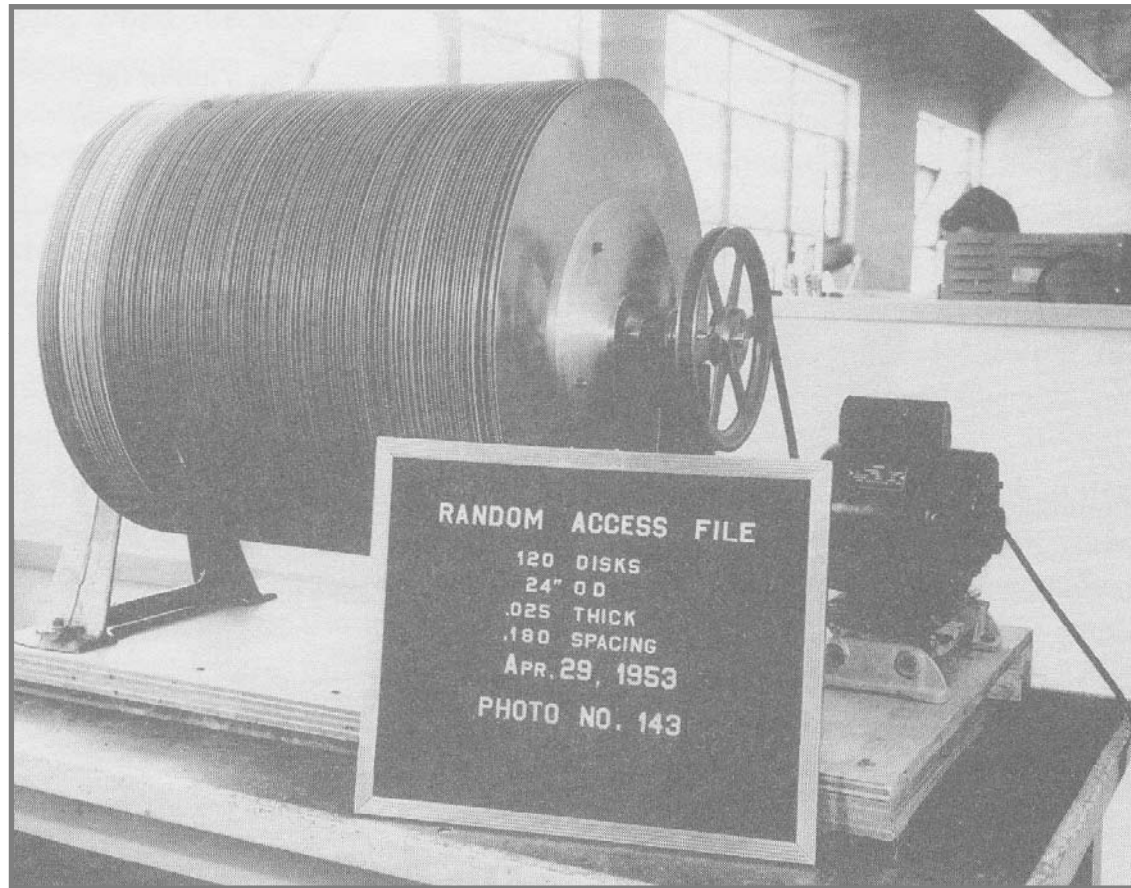
Abstraction is the key to productivity

But today's languages are no more abstract than yesterday's

The next step forward is more abstraction in data structures



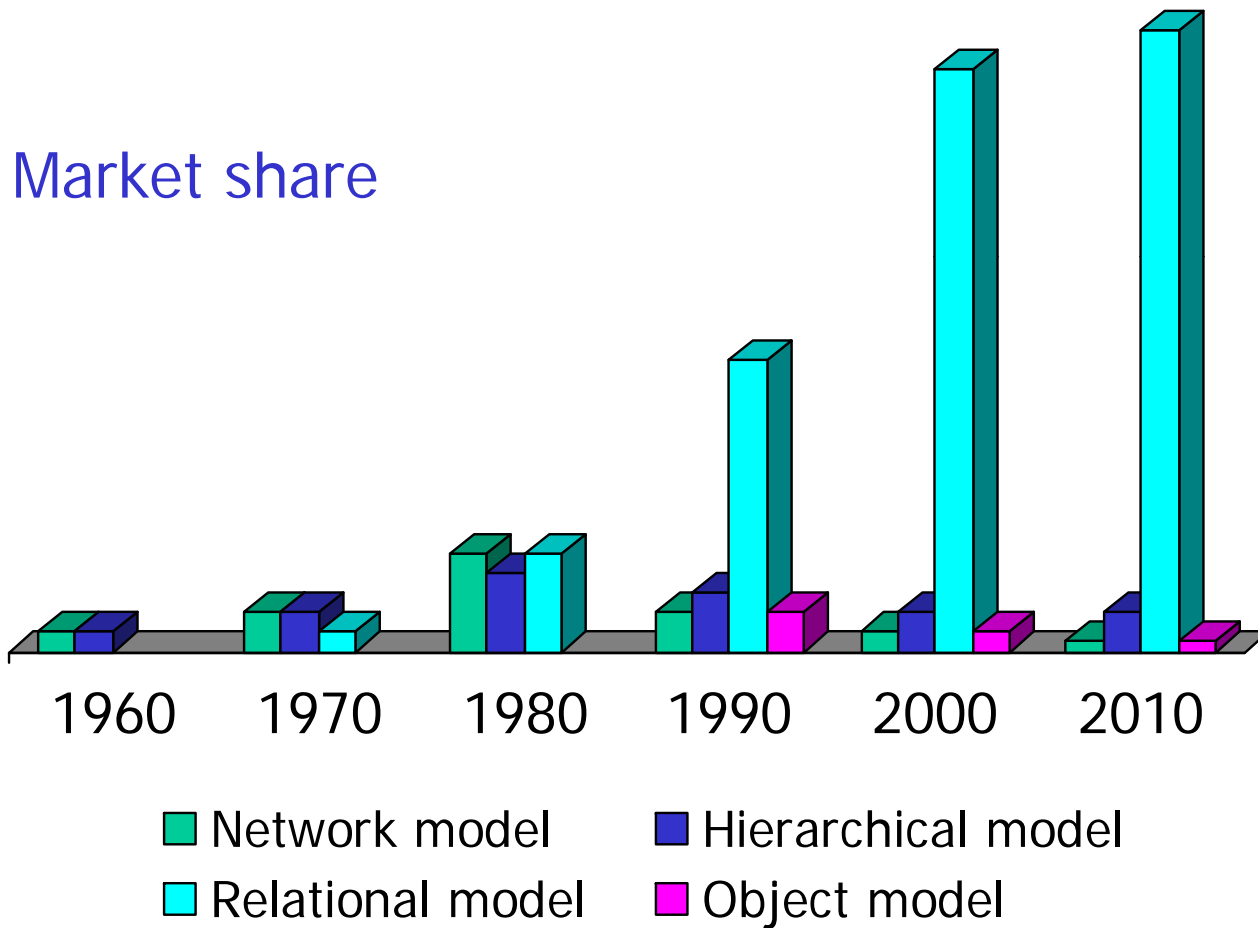
Origins of Database



April 29, 1953: The first disk drive from IBM



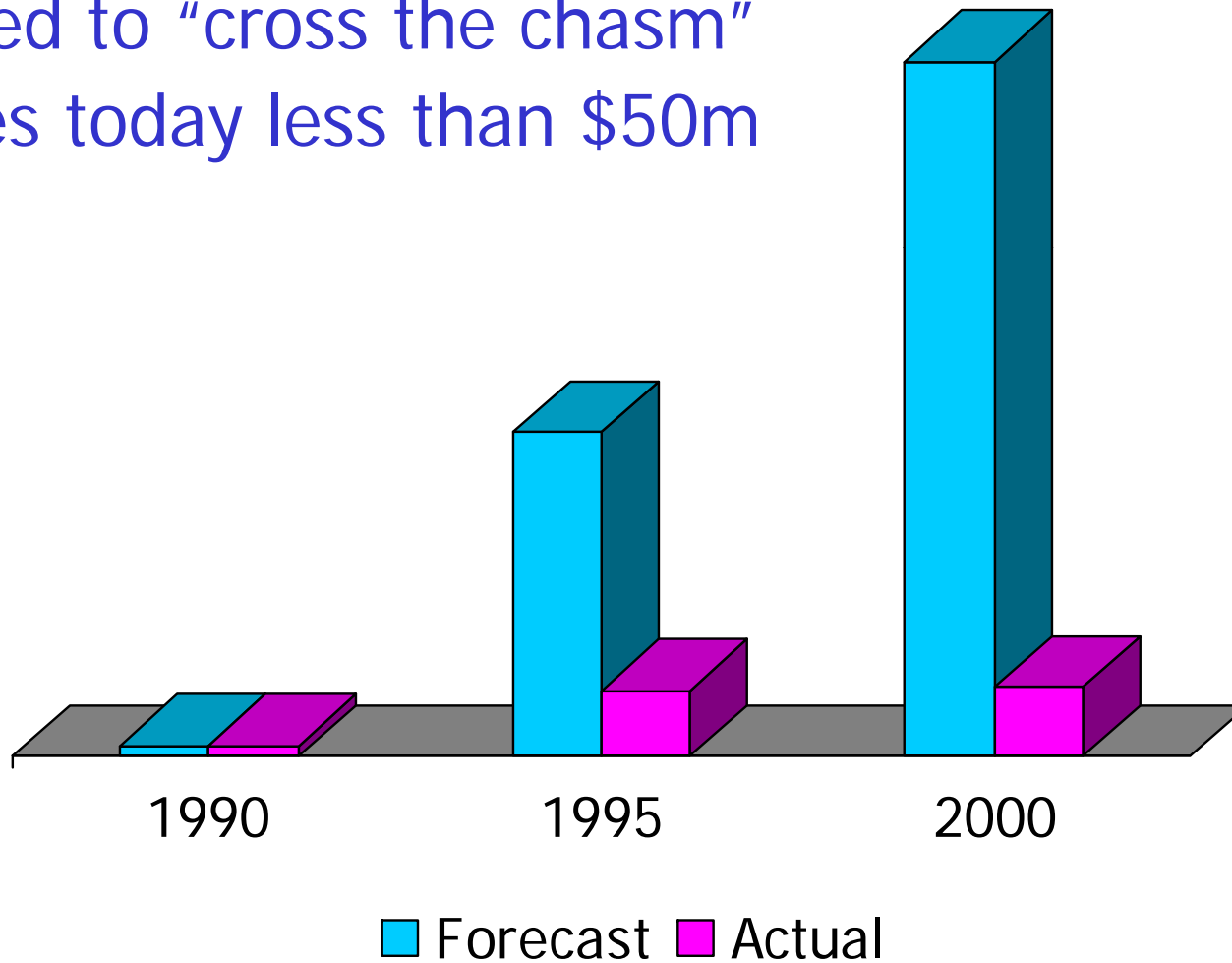
History of Data Models





Object Database

Failed to “cross the chasm”
Sales today less than \$50m





Object Model

“A storage model, not a data model”

Chris Date

- Never intended to rival relational model
- Core concepts are hostile to database



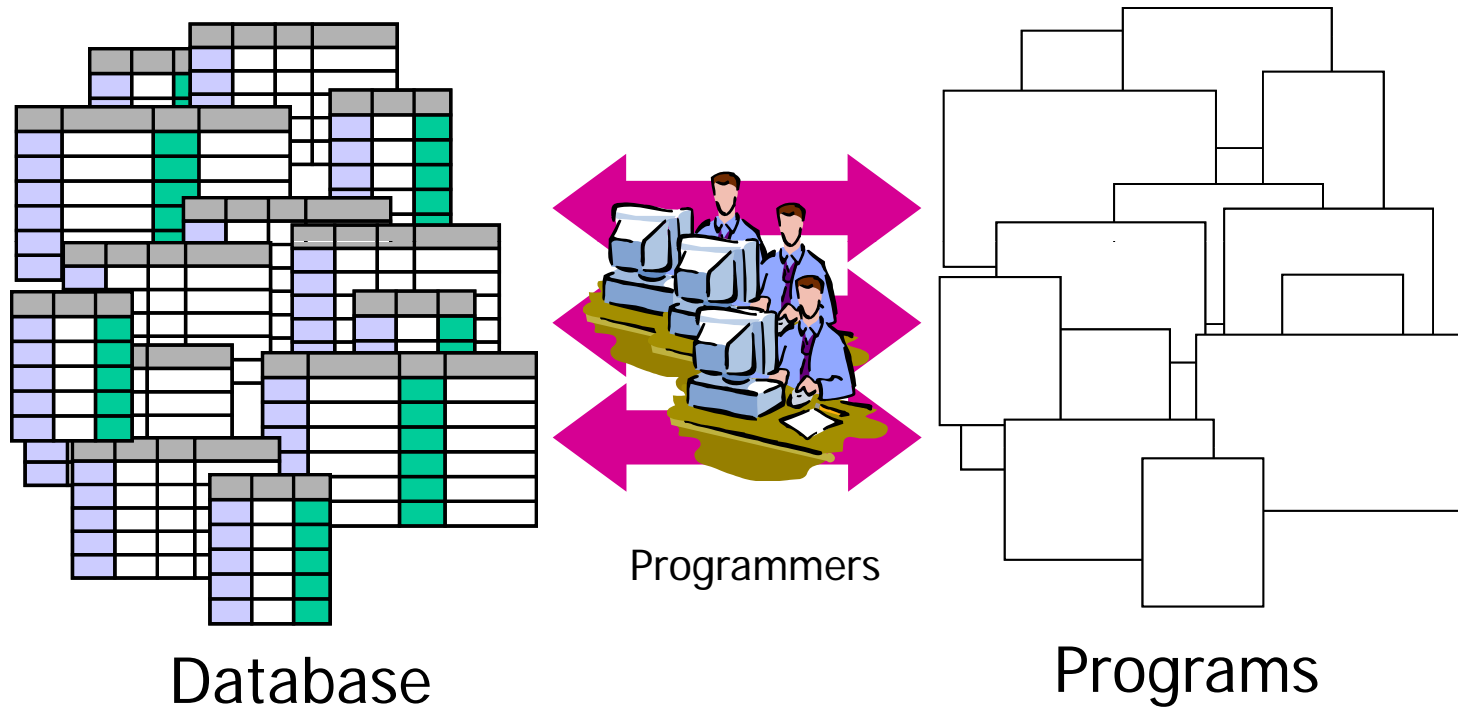
Relational Model of Data

Customer				
No	Name	Country	Credit limit	Telephone no
4555	British Airways	UK	£25,000	+44 20 7866 4455
4556	IBM	USA	\$100,000	+1 215 445 6868
4557	BP	UK	£25,000	+44 1628 664455
4558	BMW	GMY	€25,000	+49 211 80000
4559	Unilever	UK	£50,000	+44 20 8444 3333
4560	Yamaha	JPN	¥5,000,000	+91 2 400 500 60
4561	Order		Country	
4562	Code	Customer	Code	Name
	02567	4559	UK	United Kingdom
	02568	4562	USA	United States
	02569	4555	GMY	Germany
	02570	4578	JPN	Japan

Models data as tables ("relations")
One table for each different type of data



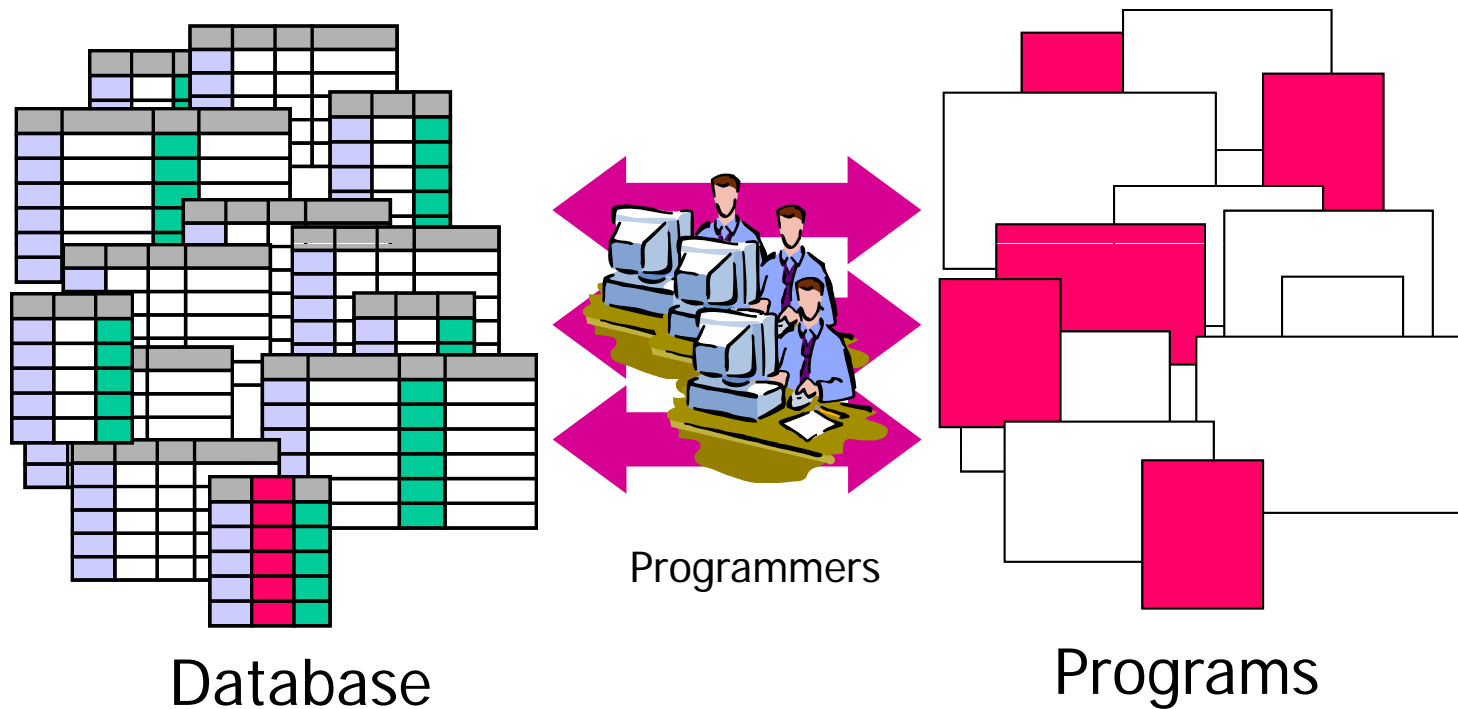
Relational Limitations



Programs are built around the data
New applications need new programs



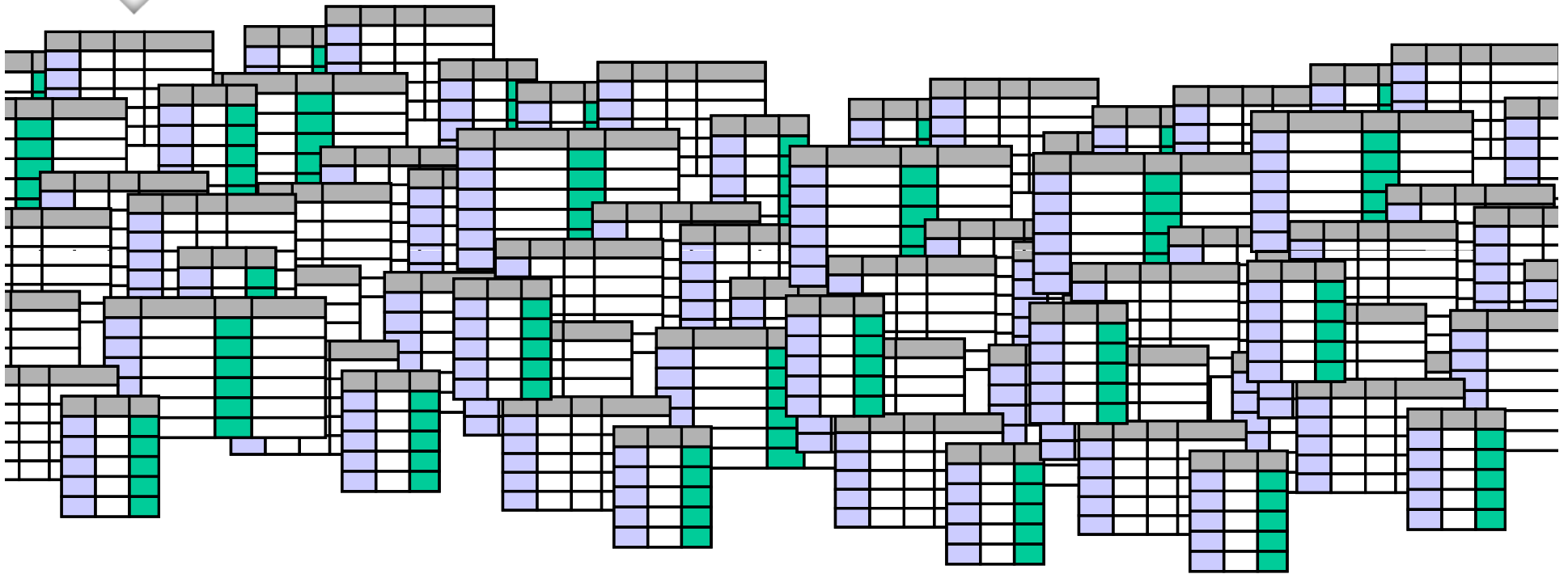
Relational Limitations



When the data structure changes,
programs have to change too



Complexity



More than 16,500 tables in SAP



Simplicity

“Everything should be made as simple as possible, but not one bit simpler”

Albert Einstein

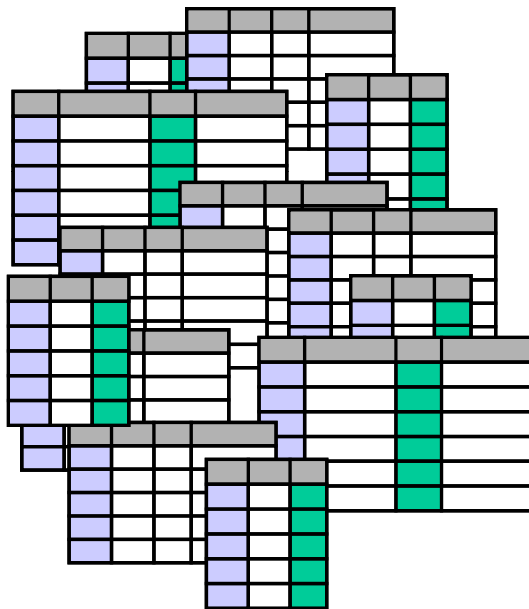
“Simplicity is complexity resolved”

Constantin Brancusi

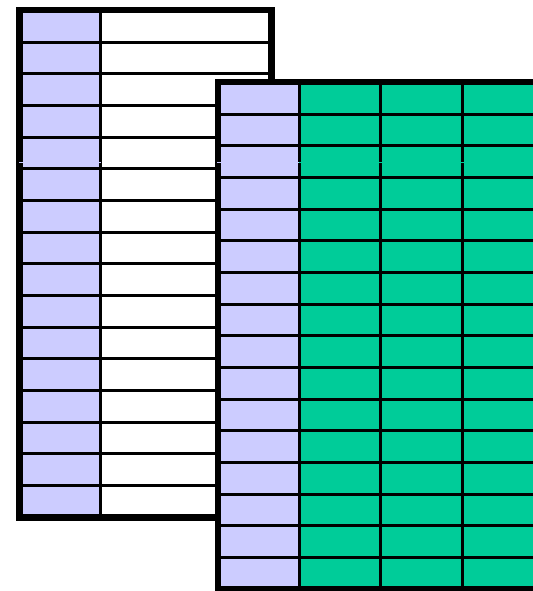


The Associative Model of Data

Relational Model



Associative Model



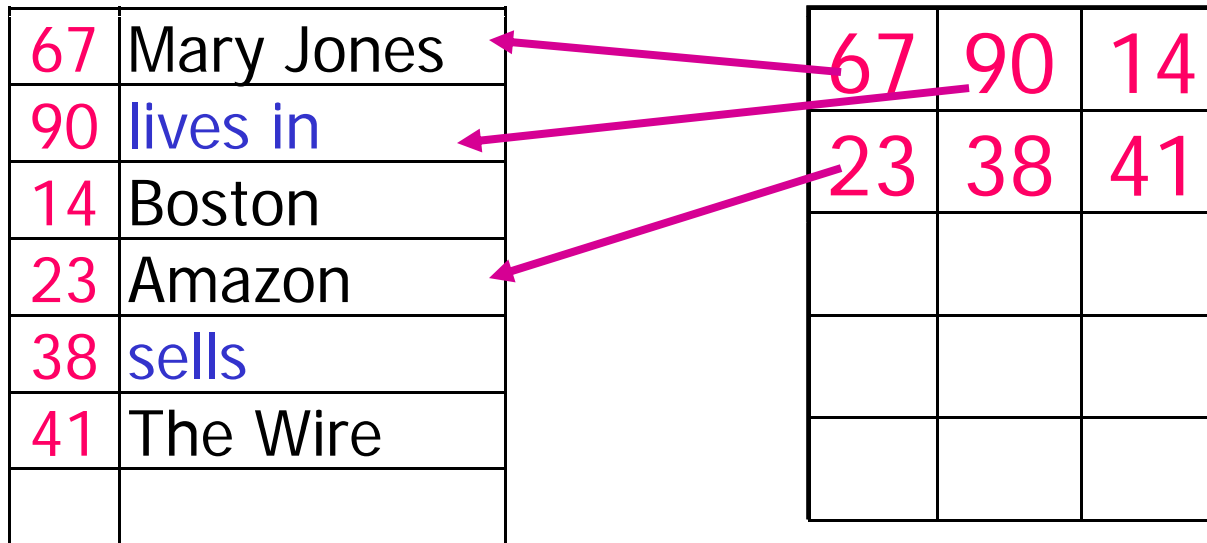
When the data structures change,
the physical structure stays the same



Triple Store

Mary Jones **lives in** Boston

Amazon **sells** The Wire



Simple, but semantically limited



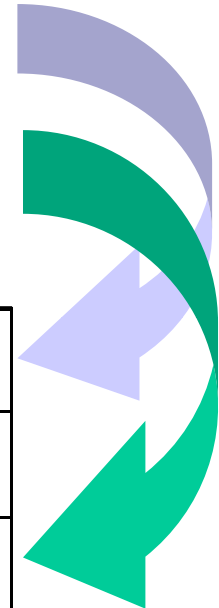
Associative Model

Mary Jones **lives in** Boston

Amazon **sells** The Wire, **for** \$9.50

67	Mary Jones
90	lives in
14	Boston
23	Amazon
38	sells
41	The Wire
53	for
98	\$9.50

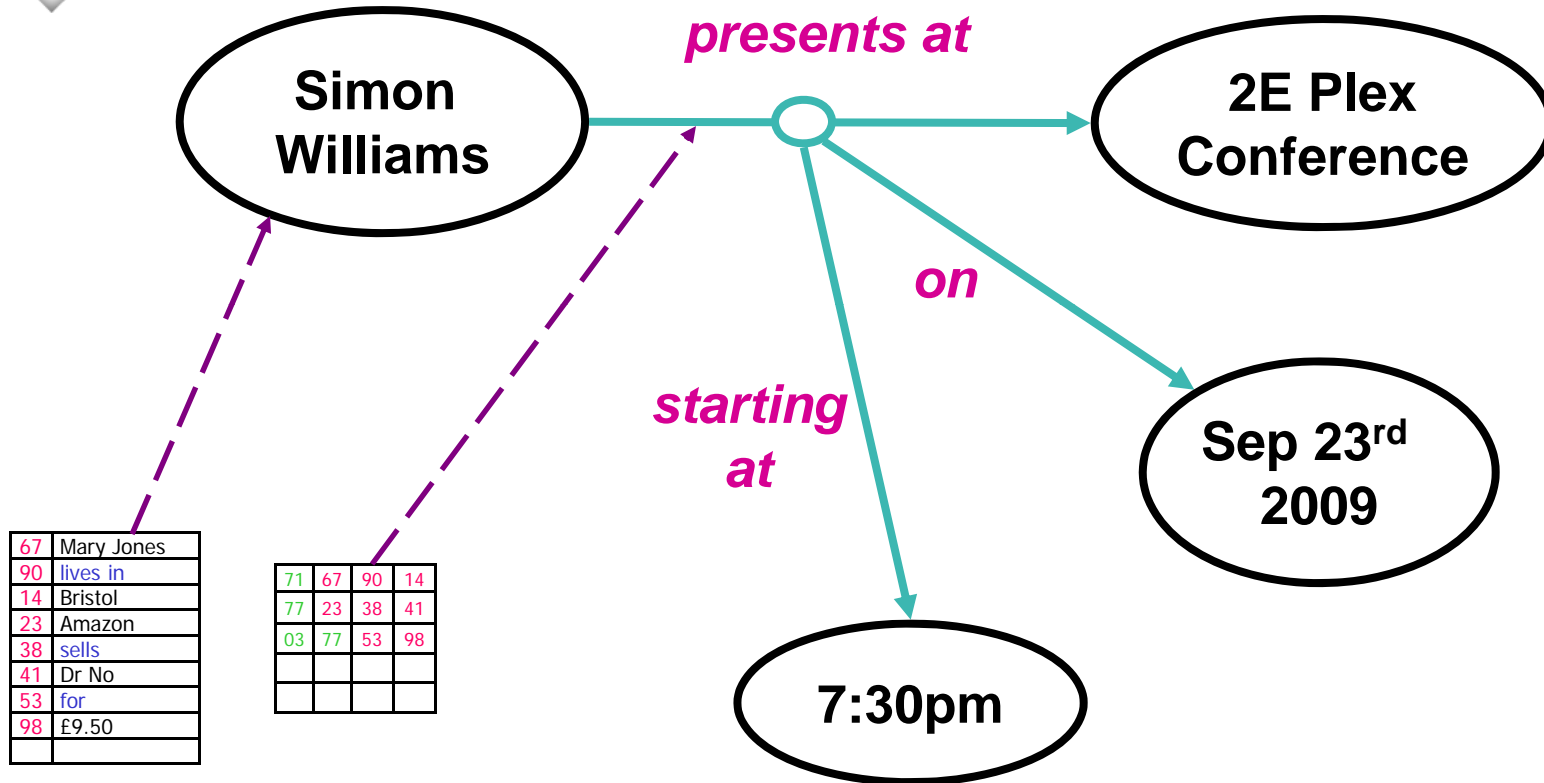
71	67	90	14
77	23	38	41
03	77	53	98



Sophisticated,
and semantically rich



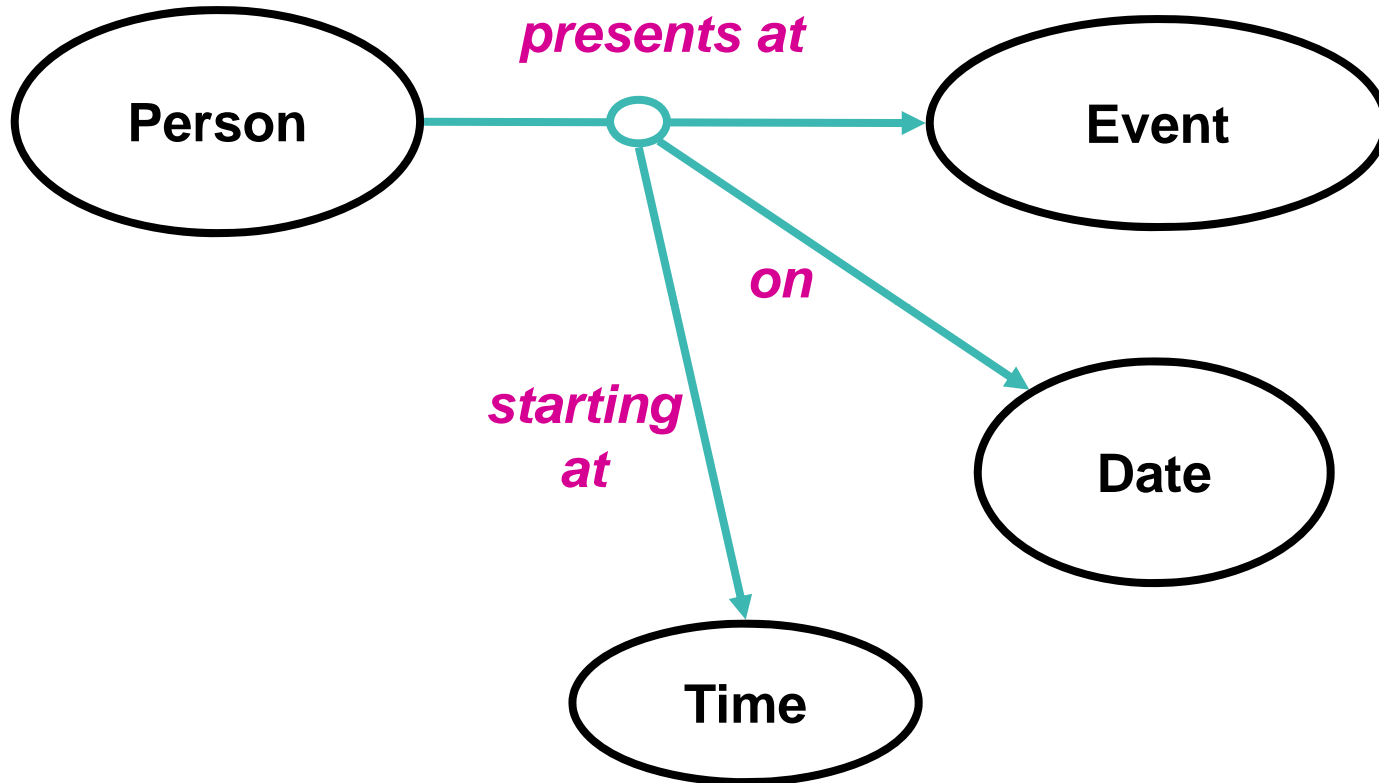
A Better Model of Reality



“Simon Williams presents at 2E Plex Conference,
... on September 23rd 2009,
... starting at 7:30pm”



Schema and Data Together



- Schema and data reside side-by-side in the database
- Both use identical concurrency mechanisms
- Schema can be altered whilst database is in use



A Better Model of Reality

- No primary or foreign keys
 - identity preserved via surrogates
- No normalisation
 - favours 1:M over M:1
- No redundancy
 - stores entities, not values
- Conceptual, logical, physical consistency
 - each entity is represented by an item on disk
 - each association is represented by a link on disk



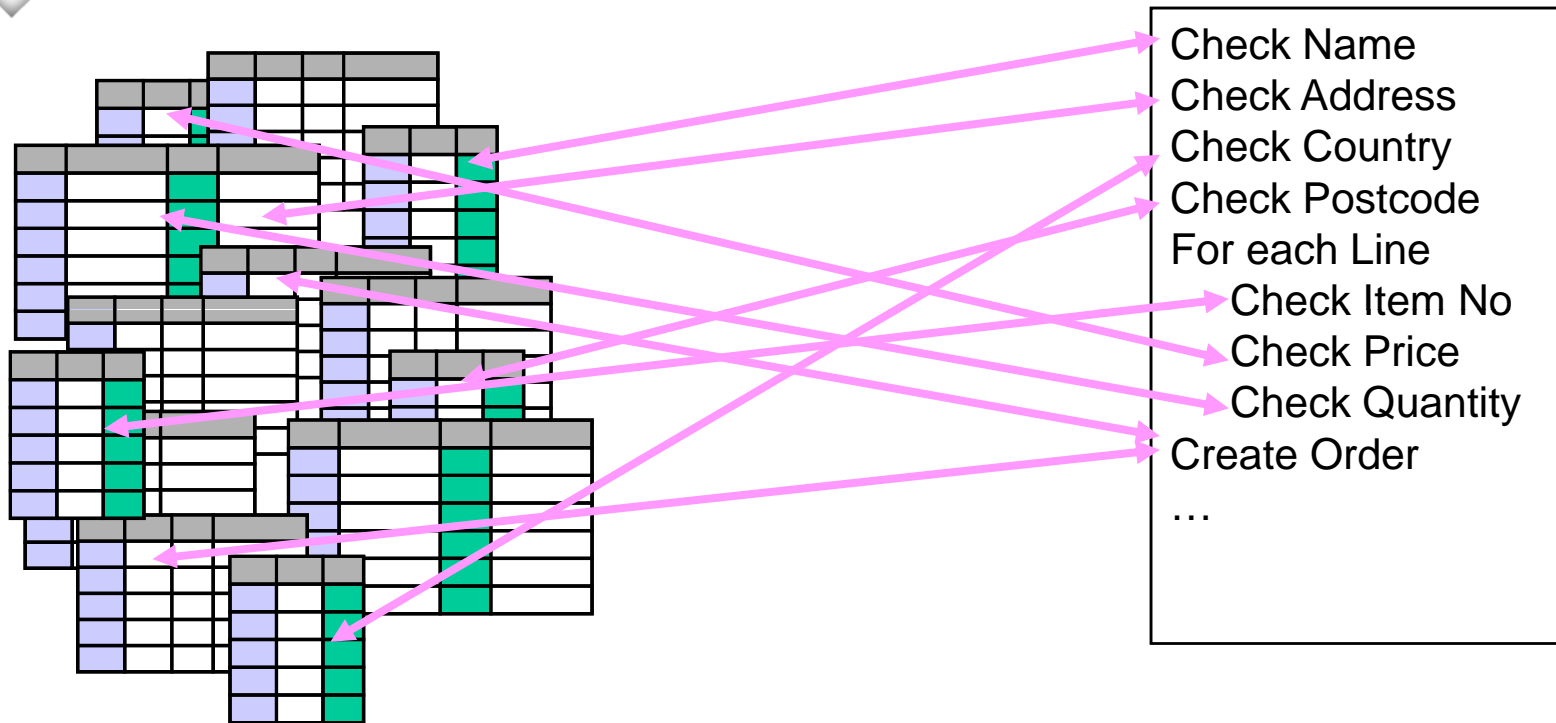
A Better Model of Reality

“Simplicity is not an end in art, but one arrives at simplicity in spite of oneself in drawing near to the reality in things.”

Constantin Brancusi



Programming the Relational Model

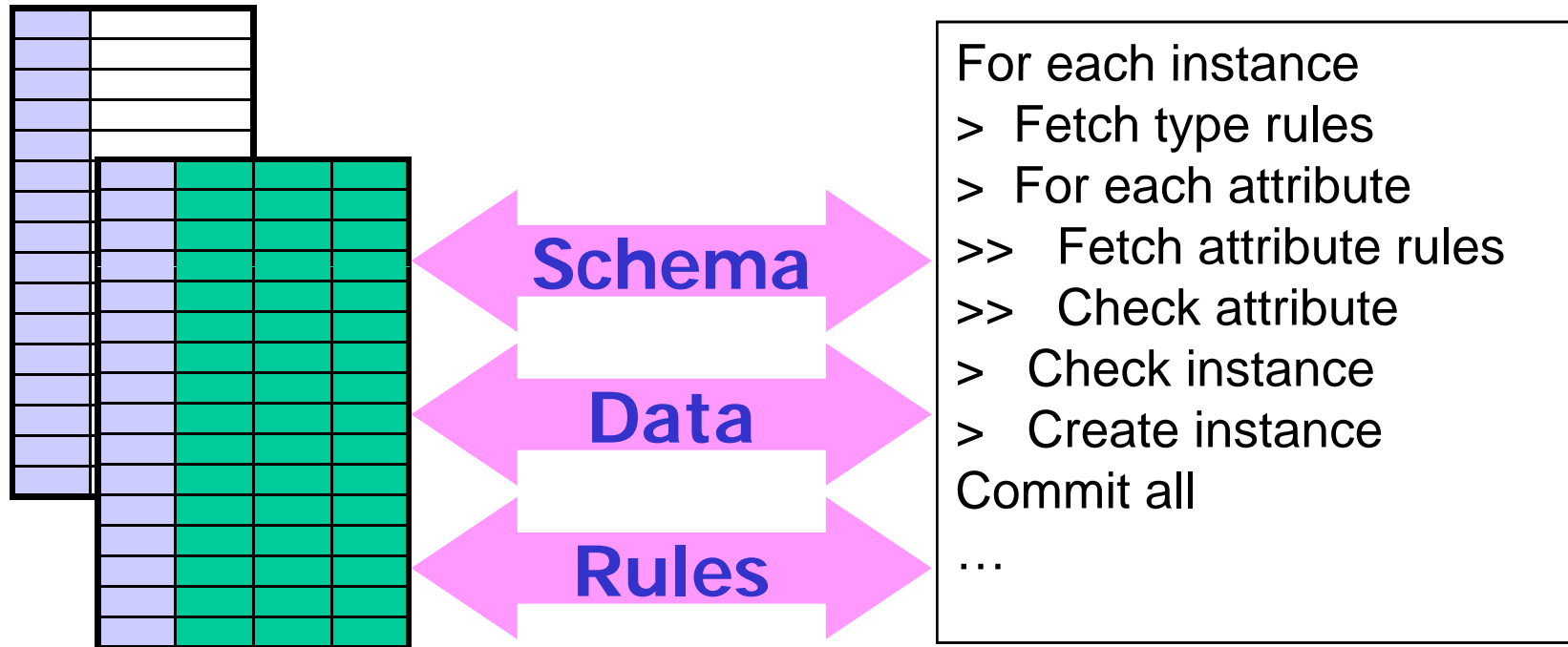


“Unicompetent programming”

Code must be unique to the tables' structure,
whether written by hand or generated



Programming the Associative Model



“Omnicompetent programming”

Code works with any data structure;
no modification, generation or compilation needed



True Program Re-use

The screenshot shows a software application with a hierarchical tree view on the left, a list of employees in the center, and a detailed view for Barney Norris in the foreground. The detailed view includes fields for Date of birth, Next of kin, Project resource?, Photograph, and Email address. A dropdown menu is open for the Next of kin field, showing a list of names including Kathleen Ferrier, Kathleen Norris, Linda Johnson, Linda LaVerde, Maluah Jackson, Marion Sinclair, Mark Goldstone, Martin Horton, Mary O'Mara, Mary Venice, Mary Williams, Maryanne Jupiter, and Matthew Duncan. Two pink arrows point from the 'Barney Norris' entry in the employee list to the 'Next of kin' dropdown menu.

Sentences Personal Edition

File Edit View Schema Query Help

Human resources Employee

Employee

- subtype of, Type
- has subset, Type
 - has subset, Current employee
 - has subset, Project resource
 - has subset, Resource skills
- date of birth, Date
- next of kin, Person
- relation to employee, Family relationship
- project resource?, Yes
- photograph, Image
- email address, Hyperlink
- event, Date
 - event, Event
 - job title, Job title
 - salary, Monetary value
 - employed by, (Group company, location, Locat

Employee

- Barney Norris
- Barry Squires
- Beverly Nash
- Bill Jupiter
- Carol Hoffman

Barney Norris

Person Employee Resource skills

Date of birth: 20-Sep-1967

Next of kin: Kathleen Norris

Project resource?: Yes

Photograph:

Email address: mailto:barneyn@sleepysoft.com

Event:

Date	Event	Job title	Salary	Employed by
01-Jun-1997	Joined	Software Engineer	\$47,000	(Sleepy Software Inc, location, New York Sales Off...
01-Feb-1999	Raise		\$58,500	
01-May-2000	Promoted	Senior Software Engineer	\$76,500	

Save Close



Abstraction in Data

“Future users of large data banks must be protected from having to know how the data is organized in the machine”

“A Relational Model of Data for Large Shared Data Banks”, E F Codd, 1970



Abstraction in Data

The relational model freed programmers from having to know the physical structure of data

The associative model frees them from having to know its logical structure



www.lazysoft.com

simon.williams@lazysoft.net