DIRECTIONS IN RELATIONAL DATABASE DESIGN

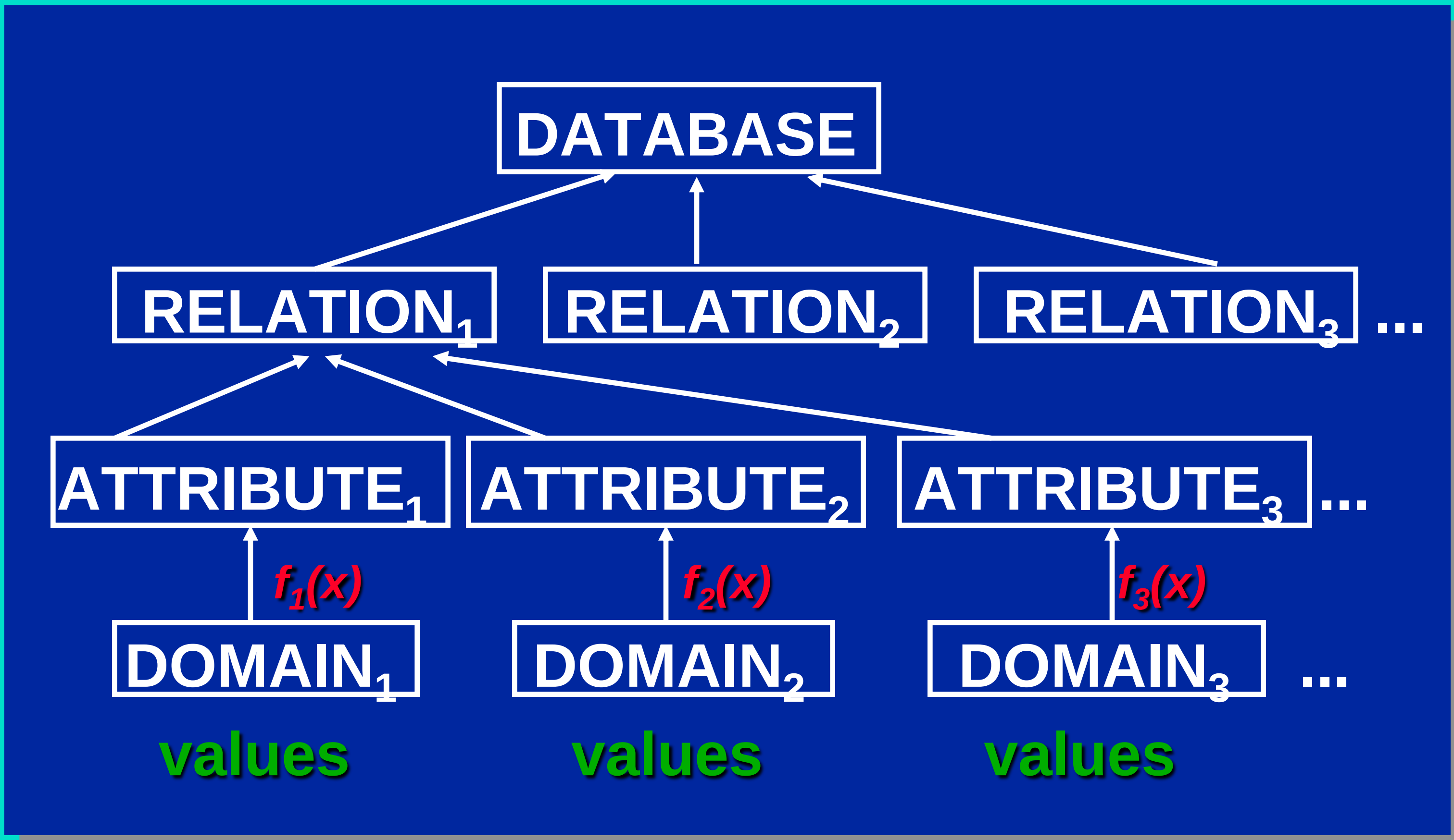
CIPS Edmonton ‘97
Edmonton, Canada
October 23, 1997
10:30 A.M.

David McGoveran
Alternative Technologies
13150 Highway 9, Suite 123
Boulder Creek, CA 95006
Telephone: 408/338-4621
www.AlternativeTech.com

OVERVIEW

- A New Understanding of Relations
- Common Design Errors
- Logical Data Independence
- Surrogate Keys
- Physical Database Design
- A New Interpretation of Normalization
- Three New Database Design Principles
- Handling Subtypes, Conditional Properties, and Conditional Relationships

CONCEPTUAL HIERARCHY OF RELATIONAL CONCEPTS



VIEWS AND LOGICAL DATA INDEPENDENCE

- Derived versus base relations: a physical notion
- In principle, there are no derived relations in the logical view
- A relation by any other name...
- Hence the importance of derived relation (not just view!) update support
- If users can't distinguish, we then have logical data independence

Products: Can't update many relations.

Theory: You can update all relations!

WHAT IS A RELATION?

- A relation is the only legitimate operand of a relational operation!
- Every result of a relational operation is a relation
- Relations represent a single type of assertion
- Each row represents a single instance of the assertion type

USE RELATION PREDICATES!

Relations should be declaratively defined by a predicate!

- **A relation predicate is a partial criteria for relation membership**
  - A FILTER FOR ROW INCLUSION
  - THE CONJUNCTION OF ALL DOMAIN, COLUMN, ROW, AND RELATION CONSTRAINTS (MULTI-RELATION CONSTRAINTS ARE EXCLUDED).
  - “THERE EXISTS AN EMPLOYEE WITH EMPLOYEE NUMBER EMP# AND NAME ENAME AND SALARY ESAL.”
- **Always specify what a relation is NOT as well!**
- **Derived relations have well-defined predicates**

COLLECTIONS OF RELATIONS

- **Types of collections:**
  - **A *DATABASE* IS A COLLECTION OF BASE RELATIONS**
  - **A *DATABASE VIEW* IS A COLLECTION OF BASE RELATIONS AND ONE OR MORE DERIVED RELATIONS**
    - » **As seen by an end-user, application, utility, developer, or transaction, etc.**
    - » **The collection is minimal with respect to its purpose: It does not include extraneous relations**
- **Collections have defining predicates**
  - **A DATABASE PREDICATE IS THE DEFINING PREDICATE FOR A DATABASE**

UNIVERSE OF DISCOURSE

- **The *universe of discourse* is defined by the database predicate.**
- **The collection of rows (each representing a fact) in a database completely define the *database state*.**
- **The difference between the *universe of discourse* and the *database state* is its *complement*.**
- **These same concepts can be applied at the relation level.**

COMMON DESIGN ERRORS

- **Using relation or attribute names improperly**
- **Using one relation for multiple entities**
- **Self-recursive relations**
- **Duplicates**
- **Under-normalization or over-normalization**
  - ***MAKES USERS JOB UNNECESSARILY COMPLEX***

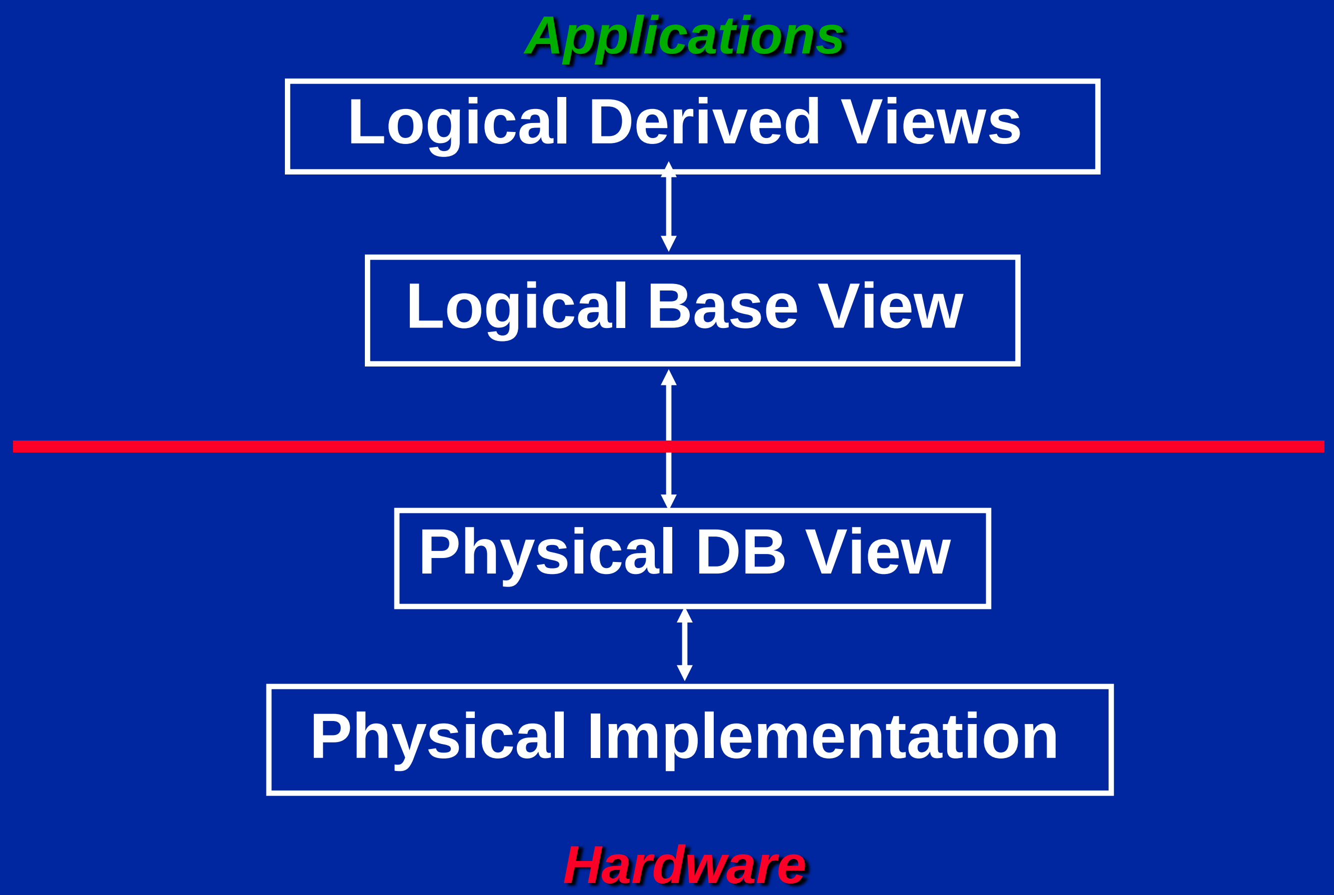
SURROGATE KEYS

Physical Implementation of a Logical Concept!

- **An artificial key, typically an integer**
- **Advantages**
  - **FASTER JOINS**
  - **SIMPLER QUERIES**
  - **SMALLER INDEXES**
  - **AVOIDS SOME NULLS (WILL EXAMINE LATER)**
  - **GUARANTEED NON-INTELLIGENT**
- **Main disadvantages**
  - **USER UNFRIENDLY**
  - **NO DIRECT VENDOR SUPPORT**

LAYERED DESIGN

The Big Picture



PHYSICAL DATABASE DESIGN

HINT: COMPILE TO THIS LAYER FOR PERFORMANCE!

- **The design of storage structures for performance!**
  - DON'T CONFUSE WITH DESIGN OF THE LOGICAL VIEW!
- **"Denormalization" (an oxymoron!) is a part of the physical database design only.**
- **Physical implementation need not be normalized, BUT...**
  - HIDE PHYSICAL DEVIATIONS FROM FROM ALL USERS
  - THE NORMALIZED LOGICAL DESIGN IS EQUIVALENT TO A SET OF UPDATABLE VIEWS OF THE PHYSICAL
  - ALL OPERATIONS (DIRECT OR INDIRECT) MANIPULATE ONLY THAT LOGICAL VIEW

DESIGN PRINCIPLES: *NORMALIZATION*

"A process by which, without loss of information, table structure is iteratively redefined so that relational operations produce expected results and only expected results." (McGoveran)

- **A fully normalized database can be viewed as one containing only relations (no tables at all)**

- **5NF is required in the logical views, BUT...**
  - **NORMALIZE RELATIVE TO CURRENT AND FUTURE APPLICATION, NOT THE ENCYCLOPEDIA**
  - **SOMETIMES FURTHER NORMALIZATION MAKES NO CHANGES**

DESIGN PRINCIPLES: NORMALIZATION

- **Some useful theorems (Date and Fagin)**
  - **BCNF AND ONE SIMPLE CK = 4NF**
  - **3NF AND ALL CKs SIMPLE = 5NF**
- **So-called “star schemas” are an ad-hoc combination of relative normalization and physical design**
- **Which collection of relations is correct?**
- **The three database design principles (applicable to any collection of relations)**

ORTHOGONALITY ***COMPLETENESS*** ***MINIMALITY***

THE PRINCIPLE OF DATABASE *ORTHOGONALITY*

- ***"In a collection of relations, every relation has a non-overlapping meaning." (Date and McGoveran)***
- **Enforce orthogonality and independence**
- **The DBMS, in principle, can determine to which relation a row belongs simply by examining its data values and data types.**
- **Demands the Information Principle as a corollary!**
  - **ALL INFORMATION IS REPRESENTED SOLELY AS VALUES IN COLUMNS**

THE PRINCIPLE OF DATABASE *ORTHOGONALITY*

- **Subtypes require special consideration**
  - TODAY'S PRODUCTS DON'T SUPPORT THEM
- **To check orthogonality of two relations R1 and R2:**
  - FORM A NEW RELATION R CONSISTING OF ALL THE ATTRIBUTES
  - ELIMINATE ANY REDUNDANT ATTRIBUTES (BE CAREFUL!)
  - IF THE R1 AND R2 HAVE ***EXACTLY*** THE DEPENDENCIES (CONSTRAINTS) OF R, AND THE COMMON ATTRIBUTES OF THE TWO RELATIONS FORM A CANDIDATE KEY OF AT LEAST ONE OF THE TWO RELATIONS, THEY ARE INDEPENDENT.

THE PRINCIPLE OF DATABASE *COMPLETENESS*

"The collection of relations in a database, along with the relational operators, is expressively complete with respect to the intended application set." (McGoveran)

- **The intended application set defines the Universe of Discourse**
  - **CURRENT APPLICATIONS**
  - **FUTURE APPLICATIONS**
- **Excludes applications, data, and data dependencies not relevant to the business**

THE PRINCIPLE OF DATABASE *MINIMALITY*

"The collection of relations in a database, along with the relational operators, permit neither statements of facts that are outside the intended application set nor redundant expressions of facts within the intended application set." (McGoveran)

- **Prevents user confusion**
- **Defines relation and database complements**
- **Prevents ill-defined database extensions**
- **Permits the closed world assumption**

CONDITIONAL DATA ENTRY WITH DEFAULTS

(HANDLING "MISSING" INFORMATION)

Use for:

- **Some conditional data entry.**
- **When the default value is meaningful or an appropriate guess!**
- **When the default value is the best estimate and otherwise harmless (i.e.., nothing depends on the particular value)**

CRITICAL ASSUMPTION:

ALL SUCH DATA IS INTENDED TO BE IMPROVED UPON OVER TIME!

CONDITIONAL RELATIONSHIPS

(HANDLING "MISSING" INFORMATION)

- **Employee-managers example**

  EMP (EMP#, ENAME, ESAL, MGR#)

- **"Approved" approach**

  EMP (EMP#, ENAME, ESAL), MGR (MGR#, ...), M_E (EMP#, MGR#)

  - ASYMMETRY PERMITS THE POSSIBILITY THAT SOME EMP# IS NOT MANAGED BY ANY MGR#

- **Recursive (cyclic) relations occur because multiple roles are represented in a single entity!**
  - A SIMILAR METHOD RESOLVES ANY N-CYCLE
- **Associate relations can model any relationship!**
- **Solves referential integrity problems ("null" FKs)**

CONDITIONAL PROPERTIES TYPES AND SUBTYPES

(HANDLING “MISSING” INFORMATION)

- **Logically, each subtype is a separate relation**
  - **“PROJECTING AWAY" A COLUMN REPRESENTS GENERALIZATION OF THE TYPE**
  - **MAKES NO STATEMENT ABOUT THE "MISSING" COLUMN!**
  - **CONVERSELY, A SUBTYPE IS A SPECIALIZATION**
  - **WORKAROUND: IMPLEMENT VIA PROJECTION VIEWS ON THE SUPERTYPE PHYSICAL RELATION**
    - » **Never let the application see columns that do not apply (don’t use nulls)**
- **Eliminates outer join, outer union, etc.**
  - **THESE CONFUSE GENERALIZATION AND PROJECTION**

UNIDENTIFIED ENTITY INSTANCES

(HANDLING "MISSING" INFORMATION)

- The unassigned employee example
  - ALWAYS REPORTS TO SOMEONE, PERHAPS FOR REASSIGNMENT
  - ALWAYS RECEIVES PAYMENT AUTHORIZATION FROM SOMEONE
- Conceptually belongs to an abstract or virtual department
  - FOR EXAMPLE, NEW HIRES
  - REPRESENTS FUNCTIONAL, THOUGH ABSTRACT BUSINESS ENTITIES
- Often modeled with null for department "value"

BENEFITS OF THE NEW DESIGN TECHNIQUES

- **The new view updating algorithms work correctly:**
  - **PREDICTABLE IMPACT OF NULLS AND DUPLICATES**
  - **UPDATABLE VIEWS CAN IMPLEMENT ANY RI ACTION**
- **Algorithms for merging multiple databases, migrating a database to relational, extending a database, etc.**
- **Design problems are identifiable/addressable**
- **Database meanings are clearer to users**
- **NULLs and three valued logic are unnecessary!**
- **Performance and development time improve**
- **BUT, nothing is guaranteed if the design is bad!**

SOME REFERENCES

- D. McGoveran, *The Client/Server University: Effective Database Design*, C. 1997 Alternative Technologies
  - A three day seminar. All the material in this presentation (and more) is covered in depth.
- C. J. Date, *Introduction to Database Systems*, 5th Edition, C. Addison Wesley
  - Brief early presentation on our work on relation predicates
- C. J. Date (and D. McGoveran), *Relatinoal Database Writings 1991-1994*, Chapters 3-5, C. 1995 Addison Wesley

  Discusses the Orthogonality Principle and View Updating

SOME REFERENCES

- D. McGoveran, *Nothing from Nothing*, Parts 1 - 4, Database Programming and Design, Dec. 1993 through March 1994, C. David McGoveran
  - An indepth discusion of the DBMS importance of classical logic, danger of many-valued logic, and how to handle “missing” information through good database design.
- Check the Web site for updates, calendar, and company information:

www.AlternativeTech.com

BIOGRAPHY

- **David McGoveran is a well-known relational database consultant and president of Alternative Technologies (Boulder Creek, CA), specialists in solving difficult relational applications problems since 1981. He publishes The Database Product Evaluation Report Series; authored (with Chris Date) A Guide to SYBASE and SQL Server; and is completing Advanced Client /Server: Design Concepts, Techniques, and Principles. Portions of this presentation are based on his workshop: Designing Effective Client/Server Applications and Databases.**

PLEASE FILL OUT YOUR EVALUATIONS...

Thank you!

