"Association"

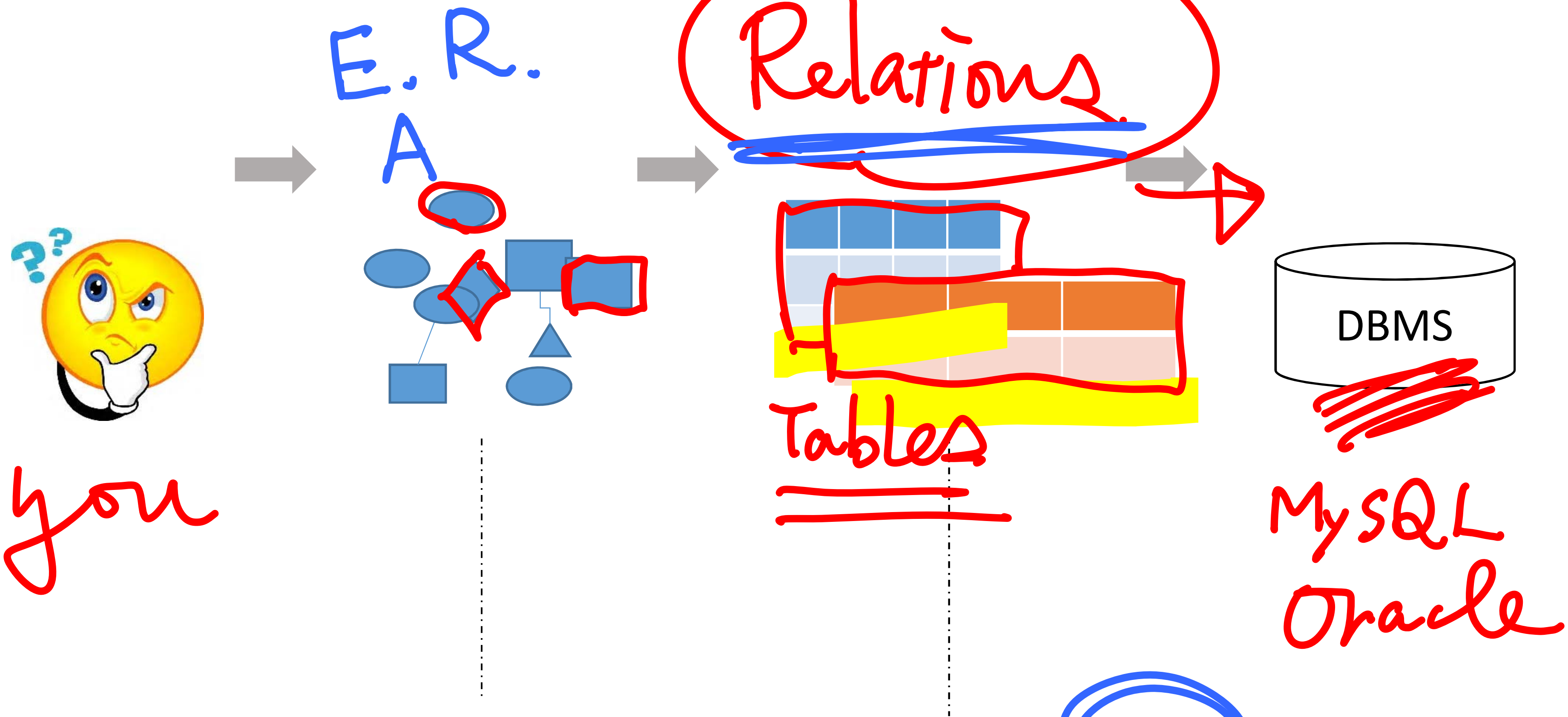
Relation al Model

CS411 Database Systems

Kevin C. Chang

Why Do We Learn This?

Database Modeling & Implementation



ER Model: Diagrams

Relational Model: Tables

ER Model vs. Relational Model

- Both are used to model data
- ER model has many concepts
  - entities, relationships, attributes, etc.
  - well-suited for capturing the app. requirements
  - not well-suited for computer implementation
  - does not even have operations on its structures
- Relational model
  - has just a single concept: relation
  - world is represented with a collection of tables
  - well-suited for efficient manipulations on computers

The Basics

"<" (1,2) (1,4)
(1,3) (2,100)
(3,5)
.....
(6,7)
(7,100)

An Example of a Relation

Table name

Products:

#1 "named" attribute

Attribute names

Name	Price	Category	Manufacturer
gizmo	$19.99	gadgets	GizmoWorks
Power gizmo	$29.99	gadgets	GizmoWorks
SingleTouch	$149.99	photography	Canon
MultiTouch	$203.99	household	Hitachi

tuples

Domains

where the values of an attr come from

- Each attribute has a type
- Must be atomic type (why? see later)
- Called *domain*
- Examples:
  - Integer
  - String
  - Real
  - …

price : float

Name : char(200)

Schemas ⟷ instance

The Schema of a Relation:

Product(Name: char(200),
Price: float)
...

- Relation name plus attribute names
  - Product(Name, Price, Category, Manufacturer)
- In practice we add the domain for each attribute

The Schema of a Database

R_1
R_2 ..

- A set of relation schemas
  - Product(Name, Price, Category, Manufacturer),
  - Vendor(Name, Address, Phone),

  ....

DB Schema
Product(- - -)
Manf, (- - -)
Person(- . -)

Instances

Actual Association of values

Prod

name	P	ma	cat
GI	19	ABC	gad ...

- **Relational schema** = R(A1,…,Ak):
  Instance = relation with k attributes (of "type" R)
  - values of corresponding domains

- **Database schema** = R1(…), R2(…), …, Rn(…)
  Instance = n relations, of types R1, R2, …, Rn

all.

Constraints are part of schema

Updates

"Basic Func"

- The database maintains a current database state.

Search

- Updates to data: Frequently – Why?
  - add a tuple
  - delete a tuple
  - modify an attribute in a tuple

Down side

- Updates to schema: Infrequently. Painful --- Why?
  - add/delete an attribute
  - add/delete a table

Data Migration

NoSQL

Think (this weekend)

Q: Schema vs. Instances

Think of it as columns vs. rows

Think of an example, answer following:

For Schema:

- When do you determine a schema?
- How often do you change your mind?

For Instance:

- When do you determine an instance?
- How often do you change your mind?

Textbook.

Behind the Scene: Database Turing Awards?

1966 A.J. Perlis

1967 Maurice V. Wilkes

1968 Richard Hamming

1969 Marvin Minsky

1970 J.H. Wilkinson

1971 John McCarthy

Network Model

1972 E.W. Dijkstra

1973 Charles W. Bachman

1974 Donald E. Knuth

1975 Allen Newell

1975 Herbert A. Simon

1976 Michael O. Rabin

1977 John Backus

1978 Robert W. Floyd

1979 Kenneth E. Iverson

1980 C. Antony R. Hoare

1981 Edgar F. Codd

1982 Stephen A. Cook

1983 Ken Thompson

1983 Dennis M. Ritchie

1984 Niklaus Wirth

1985 Richard M. Karp

1986 John Hopcroft

1986 Robert Tarjan

1987 John Cocke

1988 Ivan Sutherland

1989 William (Velvel) Kahan

1990 Fernando J. Corbato'

1991 Robin Milner

1992 Butler W. Lampson

1993 Juris Hartmanis

1993 Richard E. Stearns

1994 Edward Feigenbaum

1994 Raj Reddy

1995 Manuel Blum

1996 Amir Pnueli

1997 Douglas Engelbart

1998 James Gray

Impl. of RDB

1999 Frederick P. Brooks, Jr.

UIUC

2000 Andrew Chi-Chih Yao

Behind the Scene: It’s all about modeling

- **1973** Charles W. Bachman
- **1981** Edgar F. Codd
- **1998** James Gray

Who’s who?

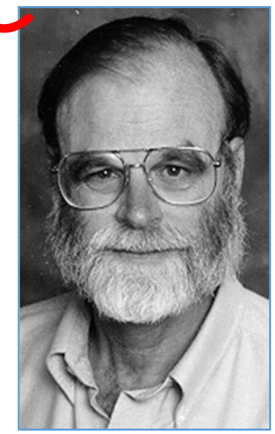
What have they contributed?

And we certainly need more!

Rel,

Codd

Realization



J.G.

Rel,

Network

Bach.

Defining DB Schema in SQL

Declaring a Relation

- Simplest form is:

```
CREATE TABLE <name> (
    <list of elements>
);
```

- And you may remove a relation from the database schema by:

```
DROP TABLE <name>;
```

- The principal element is a pair consisting of an attribute and a type.
- The most common types are:
  - INT or INTEGER (synonyms).
  - REAL or FLOAT (synonyms).
  - CHAR(*n*) = fixed-length string of *n* characters.
  - VARCHAR(*n*) = variable-length string of up to *n* characters.

Example: Create Table

```
CREATE TABLE Sells (
        bar         CHAR(20),
        beer        VARCHAR(20),
        price       REAL
);
```

Declaring Keys

- An attribute or list of attributes may be declared PRIMARY KEY or UNIQUE.

- Single-attribute keys:

```
CREATE TABLE Beers (
    name   CHAR(20) UNIQUE,
    manf   CHAR(20)
);
```

- Multi-attribute keys:

```
CREATE TABLE Sells (
        bar         CHAR(20),
        beer        VARCHAR(20),
        price    REAL,
        PRIMARY KEY (bar, beer)
    );
```

PRIMARY KEY Versus UNIQUE

- The SQL standard allows DBMS implementers to make their own distinctions between PRIMARY KEY and UNIQUE.

- Example: Some DBMS might automatically create an *index* (data structure to speed search) in response to PRIMARY KEY, but not UNIQUE.

- However, standard SQL requires these distinctions:
  - There can be only one PRIMARY KEY for a relation, but several UNIQUE attributes.
  - No attribute of a PRIMARY KEY can ever be NULL in any tuple. But attributes declared UNIQUE may have NULL's, and there may be several tuples with NULL.

Default Values

- DEFAULT <value> says that if there is no specific value known for this attribute's component in some tuple, use the stated <value>.

```
CREATE TABLE Drinkers (
    name CHAR(30) PRIMARY KEY,
    addr CHAR(50)
        DEFAULT '123 Sesame St.',
    phone CHAR(16)
);
```

- Try this. What will be the Sally tuple?

```
INSERT INTO Drinkers(name) VALUES('Sally');
```

Adding Attributes

- We may change a relation schema by adding a new attribute (“column”) by:

```
ALTER TABLE <name> ADD
        <attribute declaration>;
```

- Example:

```
ALTER TABLE Bars ADD
  phone CHAR(16) DEFAULT ‘unlisted’;
```

Deleting Attributes

- Remove an attribute from a relation schema by:

```
ALTER TABLE <name>
        DROP <attribute>;
```

- Example: we don't really need the license attribute for bars:

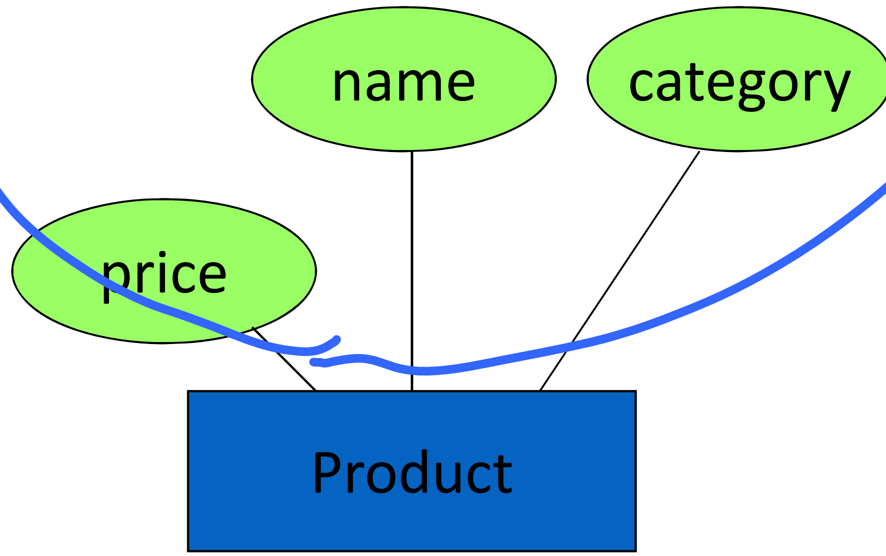
```
ALTER TABLE Bars DROP license;
```

ER to Relational Model

Basic Cases

Entity Set to Relation

Product

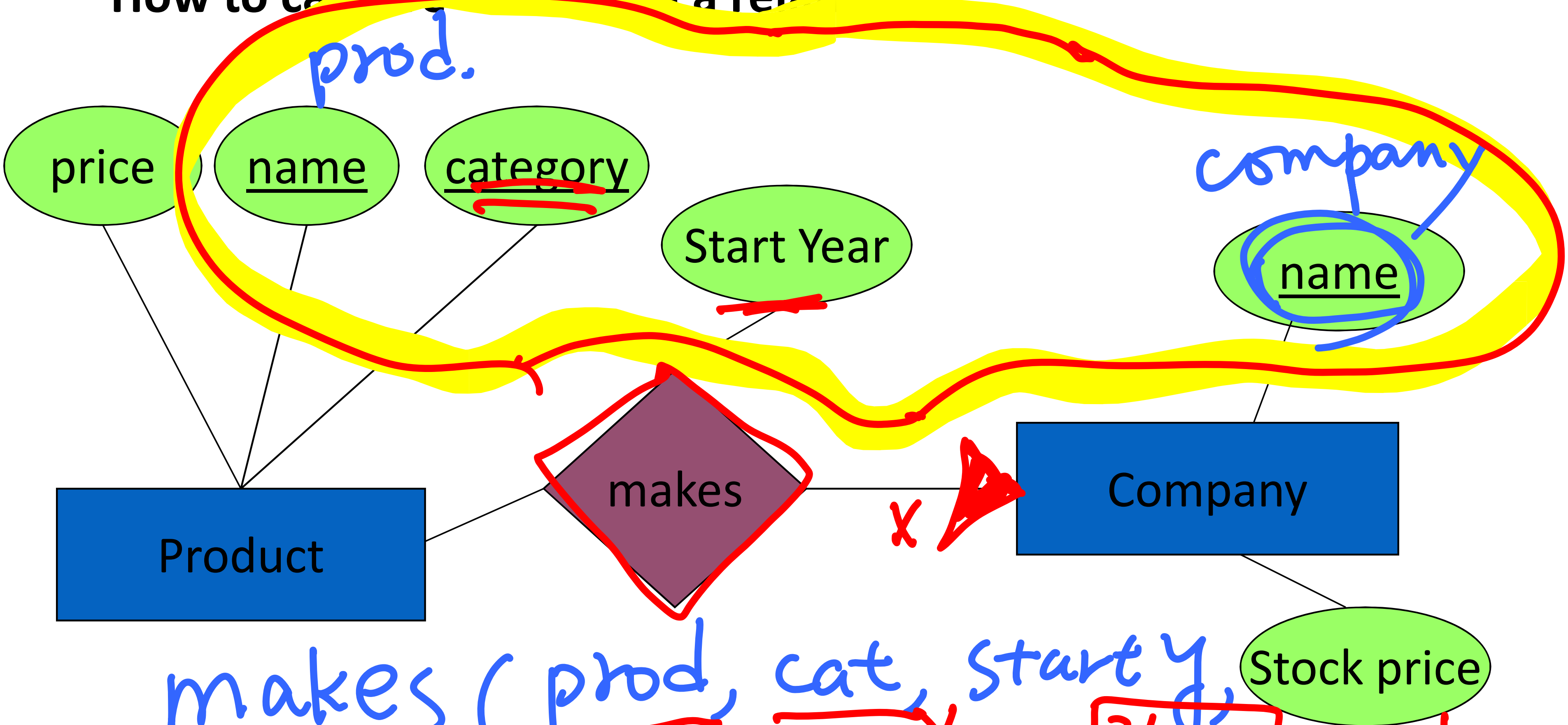


Product:

Name	Category	Price
gizmo	gadgets	$19.99

Q: Relationship → Relation?

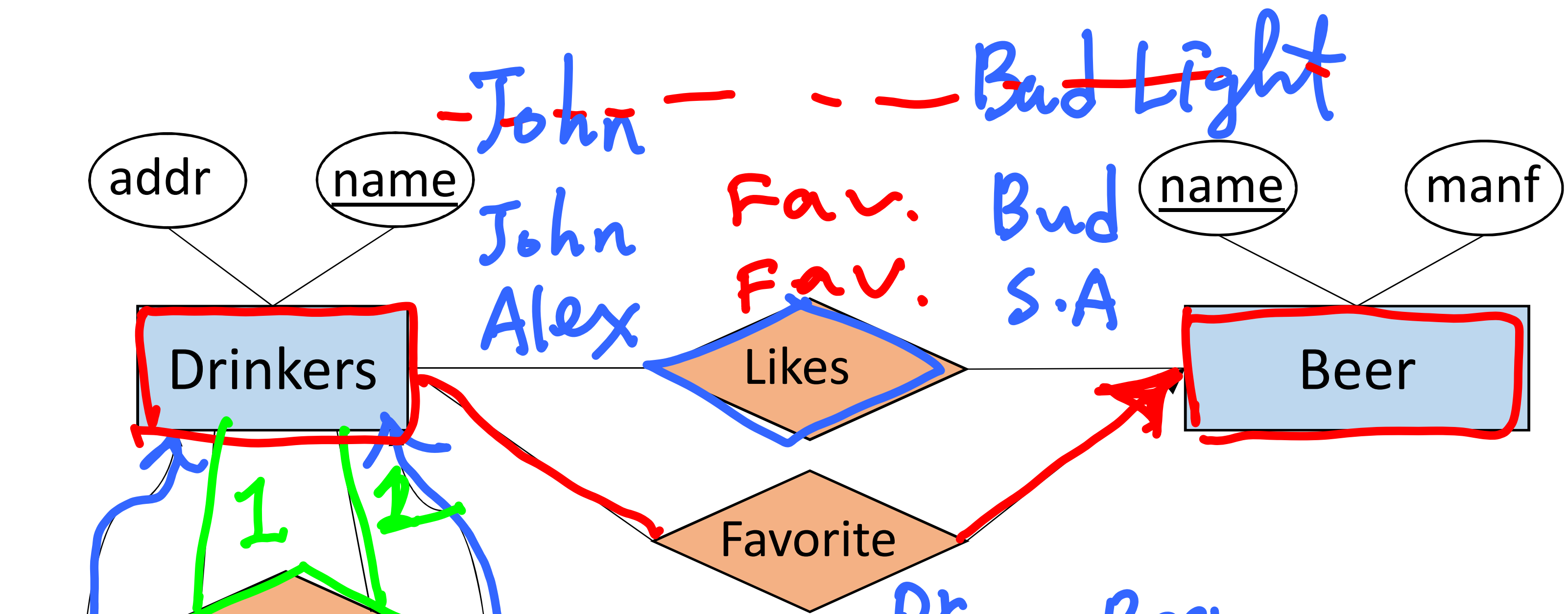
How to ca... in a relation?

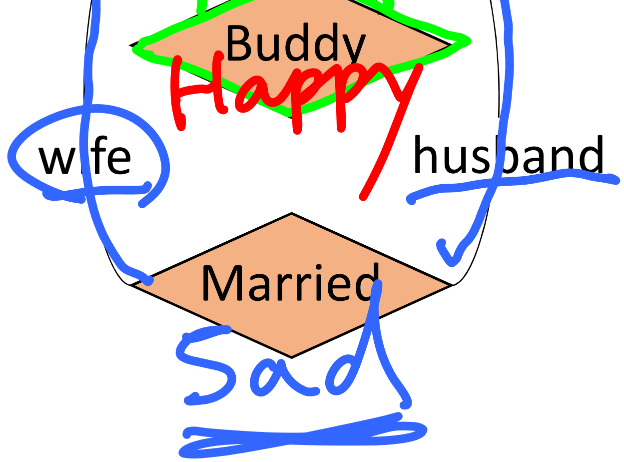


makes(prod, cat, start y, company)

$\boxed{\text{if x}}$ all three

Relationship to Relation: Another Example





- Likes(Dr_name, Beer_name)
- Favorite(drinker, beer)
- Buddy(drinker1, drinker2)
- Married(husb, wife)

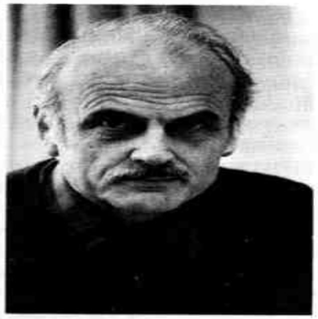
Behind the Scene:
This was how relational DBMS started…

1972 1984~

Future users of large data banks must be protected from having to know how the data is organized in the machine (the internal representation).

The 1981 ACM Turing Award Lecture

Delivered at ACM '81, Los Angeles, California, November 9, 1981



The 1981 ACM Turing Award was presented to Edgar F. Codd, an IBM Fellow of the San Jose Research Laboratory, by President Peter Denning on November 9, 1981 at the ACM Annual Conference in Los Angeles, California. It is the Association's foremost award for technical contributions to the computing community.

Codd was selected by the ACM General Technical Achievement Award Committee for his "fundamental and continuing contributions to the theory and practice of database management systems." The originator of the relational model for databases, Codd has made further important contributions in the development of relational algebra, relational calculus, and normalization of relations.

Edgar F. Codd joined IBM in 1949 to prepare programs for the Selective Sequence Electronic Calculator. Since then, his work in computing has encompassed logical design of computers (IBM 701 and Stretch), managing a computer center in Canada, heading the development of one of the first operating systems with a general multiprogramming capability, contributing to the logic of self-reproducing automata, developing high level techniques for software specification, creating and extending the relational approach to database management, and developing an English analyzing and synthesizing subsystem for casual users of relational databases. He is also the author of *Cellular Automata*, an early volume in the ACM Monograph Series.

Codd received his B.A. and M.A. in Mathematics from Oxford University in England, and his M.Sc. and Ph.D. in Computer and Communication Sciences from the University of Michigan. He is a Member of the National Academy of Engineering (USA) and a Fellow of the British Computer Society.

The ACM Turing Award is presented each year in commemoration of A. M. Turing, the English mathematician who made major contributions to the computing sciences.

Relational Database: A Practical Foundation for Productivity

E. F. Codd
IBM San Jose Research Laboratory

It is well known that the growth in demands from end users for new applications is outstripping the capability of data processing departments to implement the corresponding application programs. There are two complementary approaches to attacking this problem (and both approaches are needed): one is to put end users into direct touch with the information stored in computers; the other is to increase the productivity of data processing professionals in the development of application programs. It is less well known that a single technology, relational database management, provides a practical foundation for both approaches. It is explained why this is so.

While developing this productivity theme, it is noted that the time has come to draw a very sharp line between relational and non-relational database systems, so that the label "relational" will not be used in misleading ways. The key to drawing this line is something called a "relational processing capability."

CR Categories and Subject Descriptors: H.2.0 [**Database Management**]: General; H.2.1 [**Database Management**]: Logical Design–*data models*; H.2.4 [**Database Management**]: Systems

General Terms: Human Factors, Languages

Additional Key Words and Phrases: database, relational database, relational model, data structure, data manipulation, data integrity, productivity

Author's Present Address: E. F. Codd, IBM Research Laboratory, 5600 Cottle Road, San Jose, CA 95193.

Special Cases:
1) many one relations
2) weak entity sets
3) isa cases

Combining Relations

- It is OK to combine the relation for an entity-set *E* with the relation *R* for a many-one relationship from *E* to another entity set.

Drinker

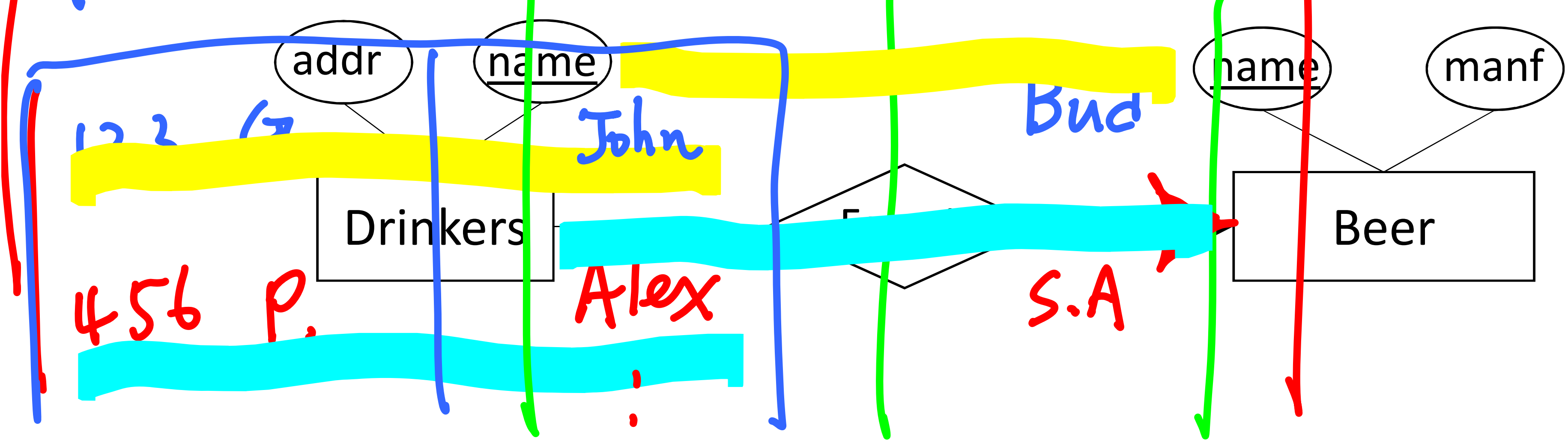
- Example:
  - Drinkers(name, addr), Favorite(drinker, beer)

  →

  - Drinker1(name, addr, favoriteBeer).

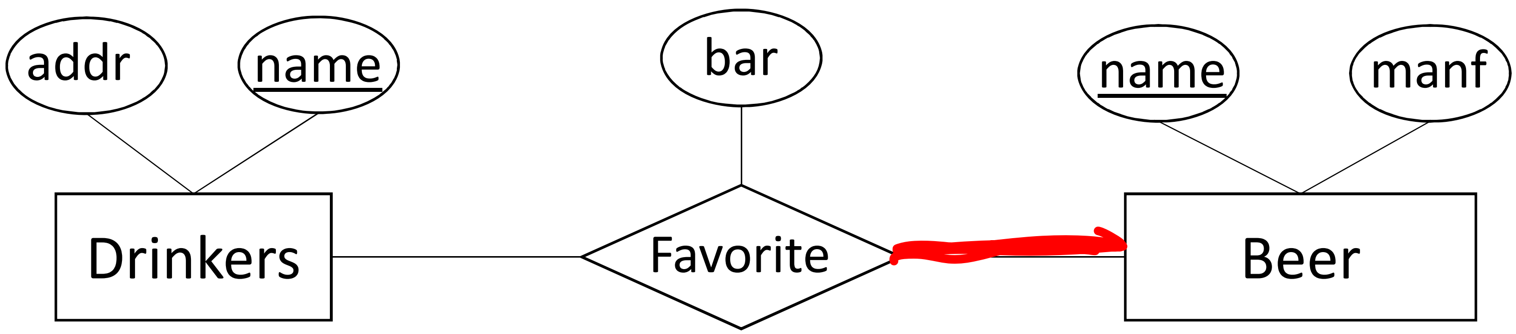
Drinker.

Fav.



Can We Merge Many-to-Many?

Not a good idea!?



John Bud

John Bud Light

Alex S.A.