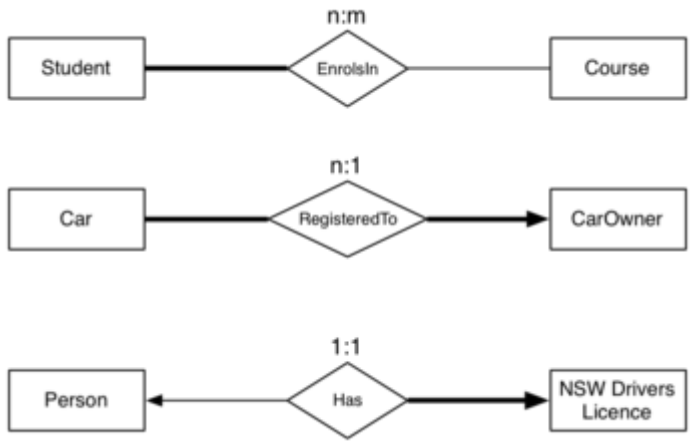
COMP3311 Week 02 Lecture

ER Model (cont)

Exercise: Relationship Types

Describe the precise semantics of each of the following:



Exercise: Medical Information

Develop an ER design for the following scenario:

- Patients are identified by an SSN, and their name, address and date of birth must be recorded.
- Doctors are identified by an SSN. For each doctor, we record their name, address, specialties and year of graduation.
- A pharmacist is identified by an SSN, and we also record their name, address and qualification.
- A medical professional is either a doctor or pharmacist, but not both. Anyone can be a patient.
- For each drug, the trade name and formula must be recorded.
- Every patient has a primary physician. Every doctor has at least one patient.
- Each pharmacy has a name, address and phone number. A pharmacy must have a manager (who is a pharmacist). A pharmacist cannot work in or manage more than one pharmacy.
- Each pharmacy sells several drugs, and has a price for each. A drug could be sold at several pharmacies, and the price could vary between pharmacies.
- Doctors prescribe drugs for patients. A doctor could prescribe one or more drugs for several patients, and a patient could obtain prescriptions from several doctors. Each prescription has a date and quantity associated with it.

[Solution]

Exercise: Book Publishing

Develop an ER design for the following scenario:

- for each person, we need to record their tax file number (TFN), their real name, and their address
- authors write books, and may publish books using a "pen-name" (a name, different to their real name, which they use as author of books)
- editors ensure that books are written in a manner that is suitable for publication
- every editor works for just one publisher
- editors and authors have quite different skills; someone who is an editor cannot be an author, and vice versa
- a book may have several authors, just one author, or no authors (published anonymously)
- every book has one editor assigned to it, who liaises with the author(s) in getting the book ready for publication
- each book has a title, and an edition number (e.g. 1st, 2nd, 3rd)
- each published book is assigned a unique 13-digit number (its ISBN); different editions of the same book will have different ISBNs
- publishers are companies that publish (market/distribute) books
- each publisher is required to have a unique Australian business number (ABN)
- a publisher also has a name and address that need to be recorded
- a particular edition of a book is published by exactly one publisher

[Solution]

Summary of ER

ER model is popular for doing conceptual design

- high-level, models relatively easy to understand
- good expressive power, can capture many details

Basic constructs: *entities*, *relationships*, *attributes*

Relationship constraints: *total / partial*, *n:m / 1:n / 1:1*

Other constructs: *inheritance hierarchies*, *weak entities*

Many notational variants of ER exist
(especially in the expression of constraints on relationships)

Relational Data Model

Relational Data Model

The *relational data model* describes the world as

- a collection of inter-connected *relations* (or *tables*)

Goal of relational model:

- a simple, general data modelling formalism
- which maps easily to file structures (i.e. implementable)

Relational model has two styles of terminology:

- mathematical: relation, tuple, attribute, ...
- data-oriented: table, record, field/column, ...

Warning: textbooks alternate between the two; treat them as synonyms.

... Relational Data Model

The relational model has one structuring mechanism ...

- a *relation* corresponds to a mathematical "relation"
- a *relation* can also be viewed as a "table"

Each *relation* (denoted *R,S,T,...*) has:

- a *name* (unique within a given database)
- a set of *attributes* (which can be viewed as column headings)

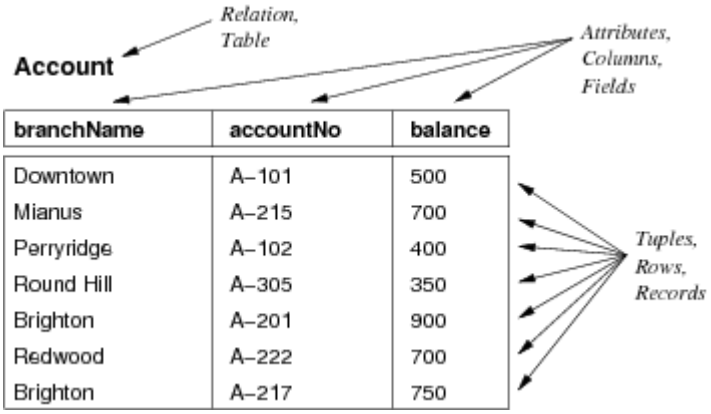
Each *attribute* (denoted *A,B,...* or $a_1,a_2,...$) has:

- a *name* (unique within a given relation)
- an associated *domain* (set of allowed values)

DB definition also uses *constraints* (logic expressions)

... Relational Data Model

Example relation (bank accounts):



... Relational Data Model

Points to note ...

Attribute values are *atomic* (no composite or multi-valued attributes).

A distinguished value `NULL` belongs to all domains.

- `NULL` has several interpretations: none, don't know, irrelevant

Each relation has a *key* (subset of attributes unique for each tuple)

... Relational Data Model

Consider relation *R* with attributes $a_1, a_2, \ldots a_n$

Relation schema of *R* : $\boldsymbol{R}(a_1{:}D_1, a_2{:}D_2, \ldots a_n{:}D_n)$

Tuple of *R* : an element of $D_1 \times D_2 \times \ldots \times D_n$ (i.e. list of values)

Instance of *R* : subset of $D_1 \times D_2 \times \ldots \times D_n$ (i.e. set of tuples)

Database schema : a collection of relation schemas.

Database (instance) : a collection of relation instances.

Database Management Systems

The *relational model* is a mathematical construct

- giving a representation for data structures
- with constraints as logic formulae on relations/tuples
- and an *algebra* for manipulating relations/tuples

Relational database management systems (RDBMSs)

- provide an implementation of the relational model
- using SQL as language for: data definition, query, update

Some approximations made by SQL:

- relations are not required to have a key
- relations are bags rather than sets

DBMS Terminology

Many DBMSs have multiple namespaces:

- DBMS-level ... database names must be unique
- database-level ... schema names must be unique
- schema-level ... table names must be unique

- table-level ... attribute names must be unique

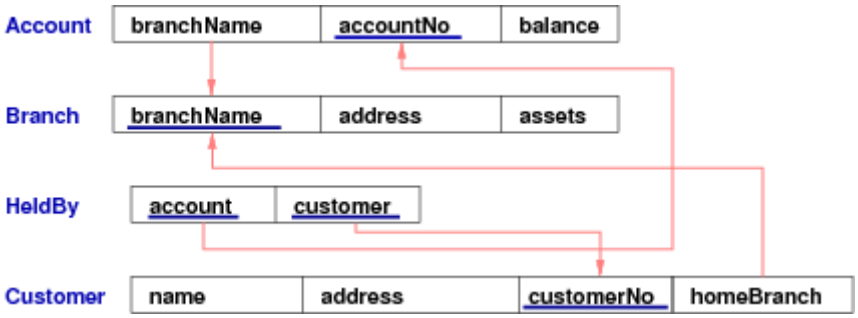
Sometimes it is convenient to use same name in several tables.

We distinguish which attribute we mean using *qualified names*

E.g. `Account.branchName` vs `Branch.branchName`

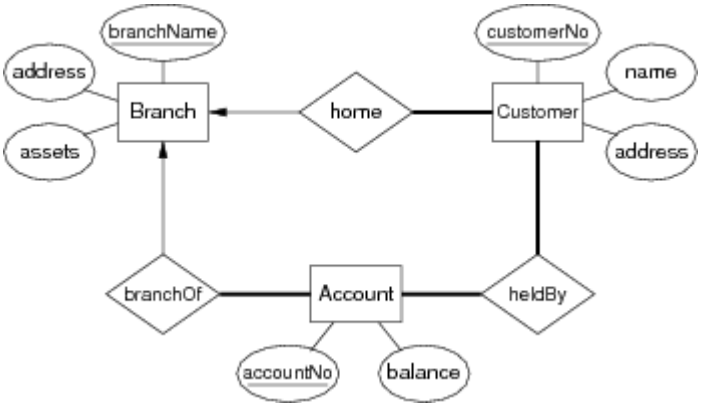
Example Database Schema

Schema with 4 relations:



... Example Database Schema

Schema is derived from the following ER design:



Example Database (Instance)

Account

branchName	accountNo	balance
Downtown	A–101	500
Mianus	A–215	700
Perryridge	A–102	400
Round Hill	A–305	350
Brighton	A–201	900
Redwood	A–222	700

Branch

branchName	address	assets
Downtown	Brooklyn	9000000
Redwood	Palo Alto	2100000
Perryridge	Horseneck	1700000
Mianus	Horseneck	400000
Round Hill	Horseneck	8000000
North Town	Rye	3700000
Brighton	Brooklyn	7100000

Customer

name	address	customerNo	homeBranch
Smith	Rye	1234567	Mianus
Jones	Palo Alto	9876543	Redwood
Smith	Brooklyn	1313131	Downtown
Curry	Rye	1111111	Mianus

Depositor

account	customer
A–101	1313131
A–215	1111111
A–102	1313131
A–305	1234567
A–201	9876543
A–222	1111111
A–102	1234567

Integrity Constraints

Relations are used to represent entities *and* relationships.

Domains limit the set of values that attributes can take.

To represent real-world problems, need to describe

- what values are/are not allowed
- what combinations of values are/are not allowed

Constraints are logical statements that do this:

- domain, key, entity integrity, referential integrity, ...

... Integrity Constraints

Domain constraints example:

- `Employee.age` attribute is typically defined as `integer`
- better modelled by adding extra constraint `(15<age<66)`

Note: `NULL` satisfies all domain constraints (except (NOT NULL))

Key constraints example:

- `Student( id, ...)` is guaranteed unique
- `Class(...,day,time,location,...)` is unique

Entity integrity example:

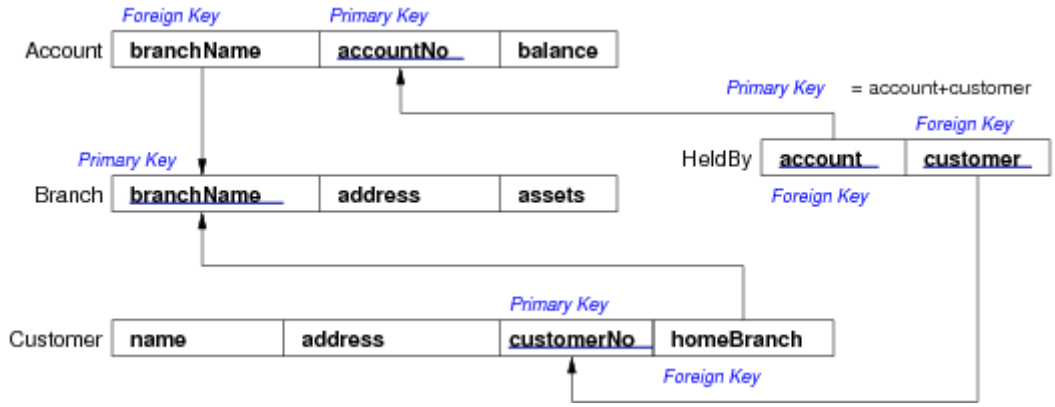
- `Class(...,Mon,2pm,Lyre,...)` is well-defined
- `Class(...,`**`NULL`**`,2pm,Lyre,...)` is not well-defined

Referential Integrity

Referential integrity constraints

- describe references between relations (tables)
- are related to notion of a *foreign key* (FK)

Example:



... Referential Integrity

A set of attributes F in relation R_1 is a *foreign key* for R_2 if:

- the attributes in F correspond to the primary key of R_2
- the value for F in each tuple of R_1
  - either occurs as a primary key in R_2
  - or is entirely `NULL`

Foreign keys are critical in relational DBs; they provide ...

- the "glue" that links individual relations (tables)
- the way to assemble query answers from multiple tables
- the relational representation of ER relationships

Relational Databases

A *relational database schema* is

- a set of relation schemas $\{ R_1, R_2, \ldots R_n \}$, and
- a set of integrity constraints

A *relational database instance* is

- a set of relation instances $\{ r_1(R_1), r_2(R_2), \ldots r_n(R_n) \}$
- where all of the integrity constraints are satisfied

One of the important functions of a relational DBMS:

- ensure that all data in the database satisfies constraints

Describing Relational Schemas

We need a formalism to express relational schemas
(which is more detailed than boxes-and-arrows diagrams used above)

SQL provides a *D*ata *D*efinition *L*anguage (DDL) for this.

```
CREATE TABLE TableName (
    attrName1 domain1 constraints1 ,
    attrName2 domain2 constraints2 ,
    ...
    PRIMARY KEY (attri,attrj,...)
    FOREIGN KEY (attrx,attry,...)
                REFERENCES
                OtherTable (attrm,attrn,...)
);
```

SQL Syntax in a Nutshell

Comments: everything after `--` *is a comment*

Identifiers: alphanumeric (a la C), but also `"An Identifier"`

Reserved words: many e.g. `CREATE`, `SELECT`, `TABLE`, ...

Strings: e.g. `'a string'`, `'don''t ask'`, but no `'\n'`

Numbers: like C, e.g. `1`, `-5`, `3.14159`, ...

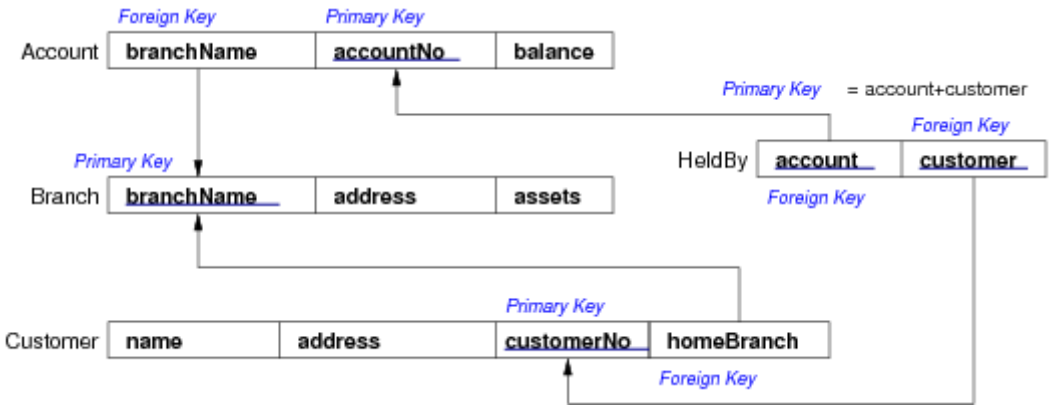
Identifiers and reserved words are case-insensitive:
`TableName` = `tablename` = `TaBLeNamE` != `"TableName"`

Types: `integer`, `float`, `char(`*n*`)`, `varchar(`*n*`)`, `date`, ...

Operators: `=`, `<>`, `<`, `<=`, `>`, `>=`, `AND`, `OR`, `NOT`, ...

Exercise: Simple Relational Schema

Express the following schema in SQL DDL:



Assume only two domains: `text` and `integer`.

[Solution]

... Exercise: Simple Relational Schema

Augment the previous schema to enforce:

- no accounts can be overdrawn
- customer numbers are seven-digit integers
- account numbers look like `A-101`, `B-306`, etc.
- the assets of a branch is the sum of the balances in all of the accounts held at that branch

Assume that all standard SQL types (domains) are available.

Add new domain to define account numbers and use it throughout

[Solution]

Mapping ER Designs to Relational Schemas

ER to Relational Mapping

One useful strategy for database design:

- perform initial data modelling using ER or OO
  (conceptual-level modelling)
- transform conceptual design into relational model
  (implementation-level modelling)

A formal mapping exists for ER model → Relational model.

This maps "structures"; but additional info is needed, e.g.

- concrete domains for attributes and other constraints

Relational Model vs ER Model

Correspondences between relational and ER data models:

- attribute(ER) ≅ attribute(Rel), entity(ER) ≅ tuple(Rel)
- entity set(ER) ≅ relation(Rel), relationship(ER) ≅ relation(Rel)

Differences between relational and ER models:

- Rel uses relations to model entities *and* relationships
- Rel has no composite or multi-valued attributes (only atomic)
- Rel has no object-oriented notions (e.g. subclasses, inheritance)

Mapping Strong Entities

An entity set *E* with atomic attributes $a_1, a_2, \ldots a_n$

maps to

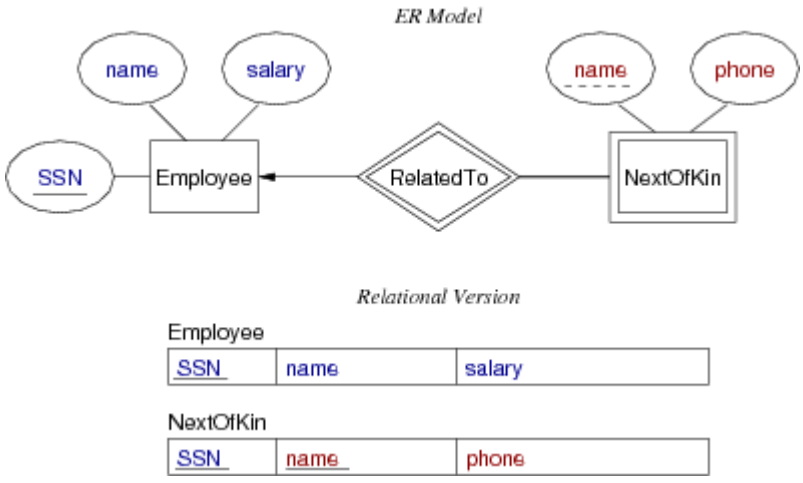
A relation *R* with attributes (columns) $a_1, a_2, \ldots a_n$

Example:

Note: the key is preserved in the mapping.

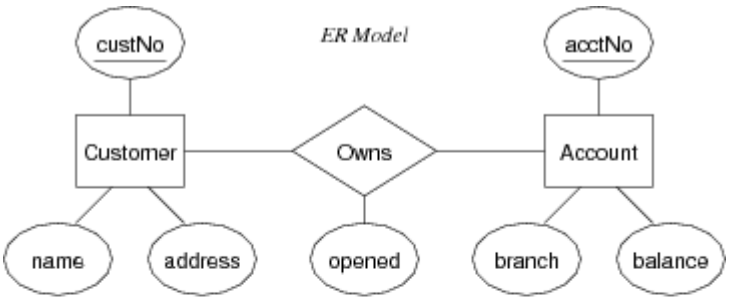
Mapping Weak Entities

Example:



Mapping N:M Relationships

Example:



Relational Version

Customer

custNo	**name**	**address**

Account

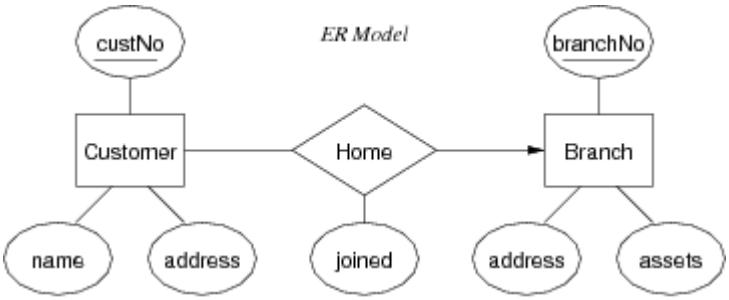
acctNo	**branch**	**balance**

Owns

custNo	**acctNo**	**opened**

Mapping 1:N Relationships

Example:



Relational Version

Customer

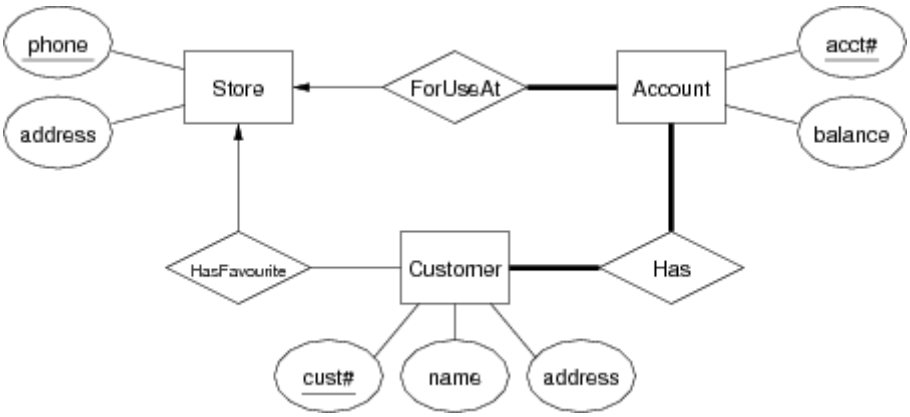
custNo	**name**	**address**	**branchNo**	**joined**

Branch

branchNo	**address**	**assets**

Exercise: ER-to-Relational (1)

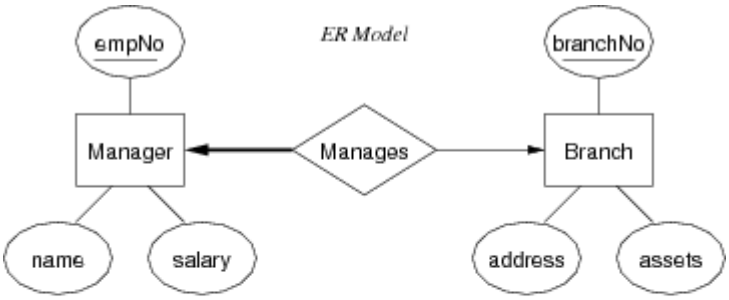
Convert this ER design to relational form:



[Solution]

Mapping 1:1 Relationships

Example:



Relational Version

Manager

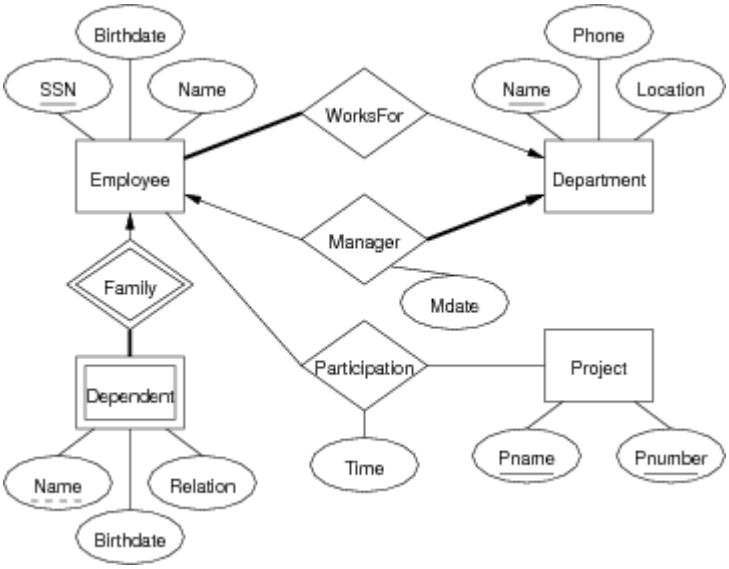
empNo	name	salary	branchNo

Branch

branchNo	address	assets

Exercise: ER-to-Relational (2)

Convert this ER design to relational form:



[Solution]

Produced: 05 March 2018