>>

Relational Model

- Relational Data Model
- Example Database Schema
- Example Database (Instance)
- Integrity Constraints
- Referential Integrity
- Relational Databases
- Describing Relational Schemas

^ >>

❖ Relational Data Model

The relational data model describes the world as

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

The relational model has one structuring mechanism: relations

- relations are used to model both entities and relationships

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)

<< ^ >>

❖ Relational Data Model (cont)

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)

Note: tuples: (2,3) ≠ (3,2) relation: { (a,b), (c,d) } = { (c,d), (a,b) }

Domains are comprised of atomic values (e.g. integer, string, date)

A distinguished value `NULL` belongs to all domains

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

<< ^ >>

❖ Relational Data Model (cont)

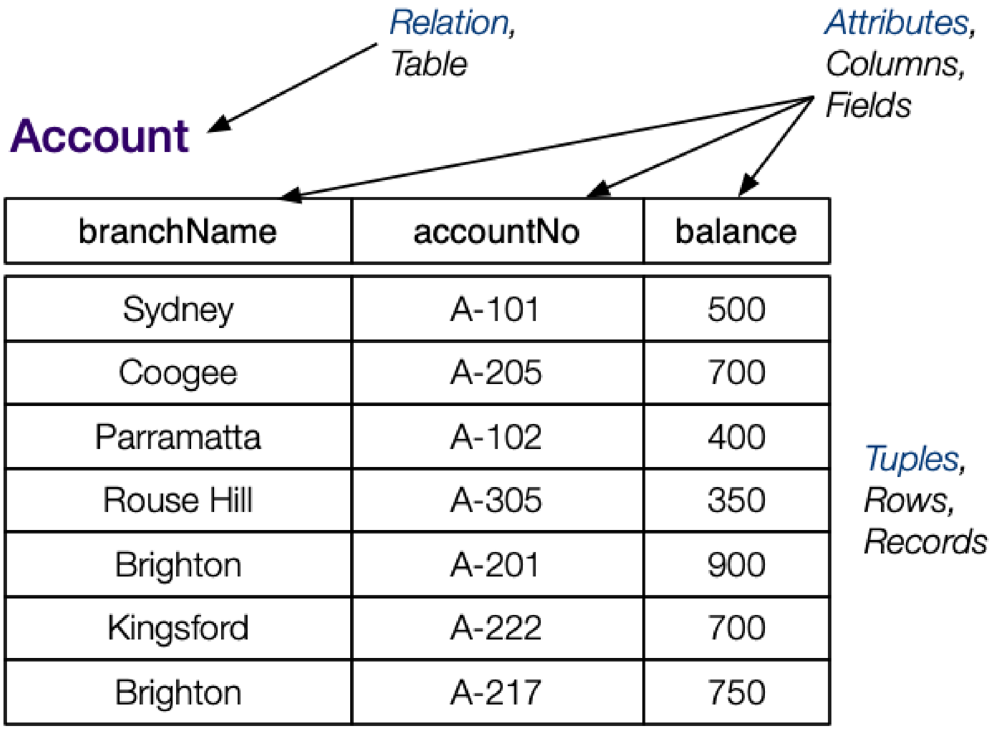
A relation: `Account(branchName, accountNo, balance)`

And an *instance* of this relation:

```
{
  (Sydney, A-101, 500),
  (Coogee, A-215, 700),
  (Parramatta, A-102, 400),
  (Rouse Hill, A-305, 350),
  (Brighton, A-201, 900),
  (Kingsford, A-222, 700)
  (Brighton, A-217, 750)
}
```

❖ Relational Data Model (cont)

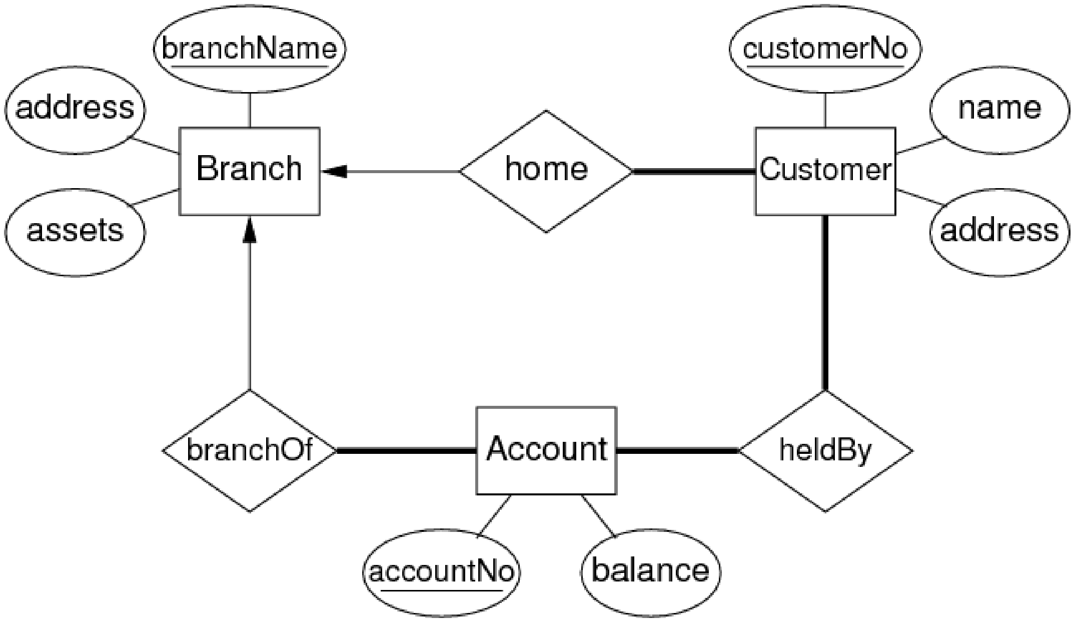
`Account` relation as a table:



branchName	accountNo	balance
Sydney	A-101	500
Coogee	A-205	700
Parramatta	A-102	400
Rouse Hill	A-305	350
Brighton	A-201	900
Kingsford	A-222	700
Brighton	A-217	750

❖ Example Database Schema

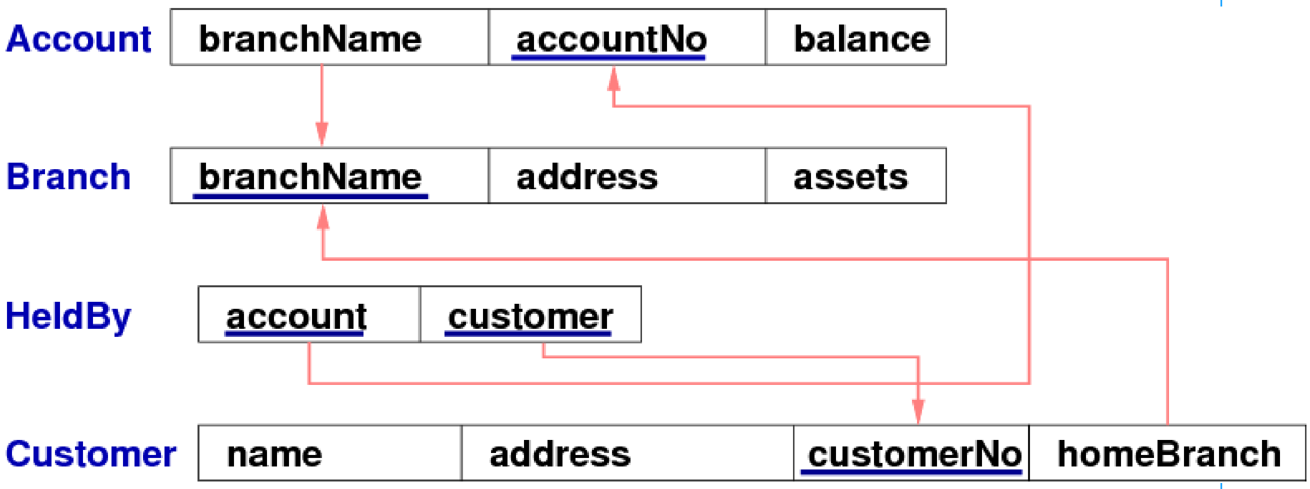
Consider the following ER data model:



<< ^ >>

❖ Example Database Schema (cont)

Relational schema derived from this ER model:



Account: branchName, accountNo, balance

Branch: branchName, address, assets

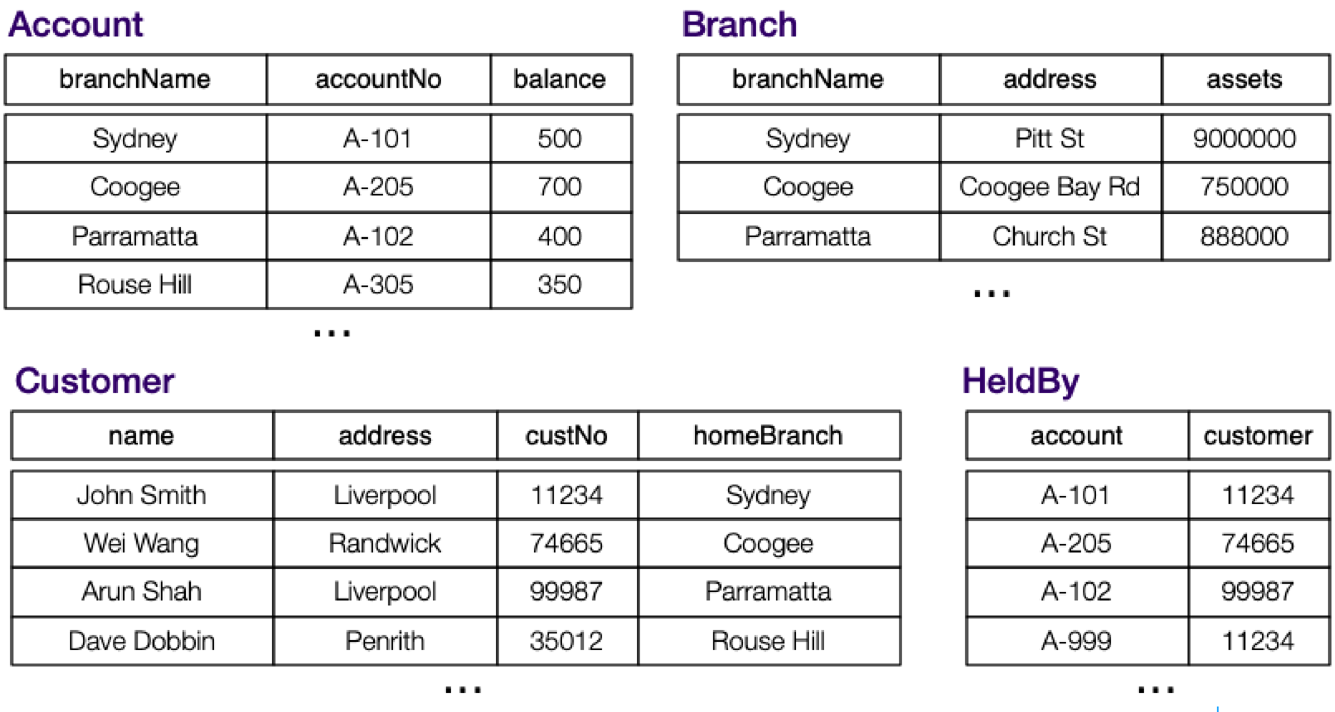
HeldBy: account, customer

Customer: name, address, customerNo, homeBranch

Note: distinguish attributes via e.g. `Branch.address` vs `Customer.address`

<< ^ >>

❖ Example Database (Instance)



Account

branchName	accountNo	balance
Sydney	A-101	500
Coogee	A-205	700
Parramatta	A-102	400
Rouse Hill	A-305	350

...

Branch

branchName	address	assets
Sydney	Pitt St	9000000
Coogee	Coogee Bay Rd	750000
Parramatta	Church St	888000

...

Customer

name	address	custNo	homeBranch
John Smith	Liverpool	11234	Sydney
Wei Wang	Randwick	74665	Coogee
Arun Shah	Liverpool	99987	Parramatta
Dave Dobbin	Penrith	35012	Rouse Hill

...

HeldBy

account	customer
A-101	11234
A-205	74665
A-102	99987
A-999	11234

...

<< ∧ >>

❖ Integrity Constraints

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 constraints:
  limit the set of values that attributes can take
- key constraints:
  identify attributes that uniquely identify tuples
- entity integrity constraints: require keys to be fully-defined
- referential integrity constraints:
  require references to other tables to be valid

<< ^ >>

❖ Integrity Constraints (cont)

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:

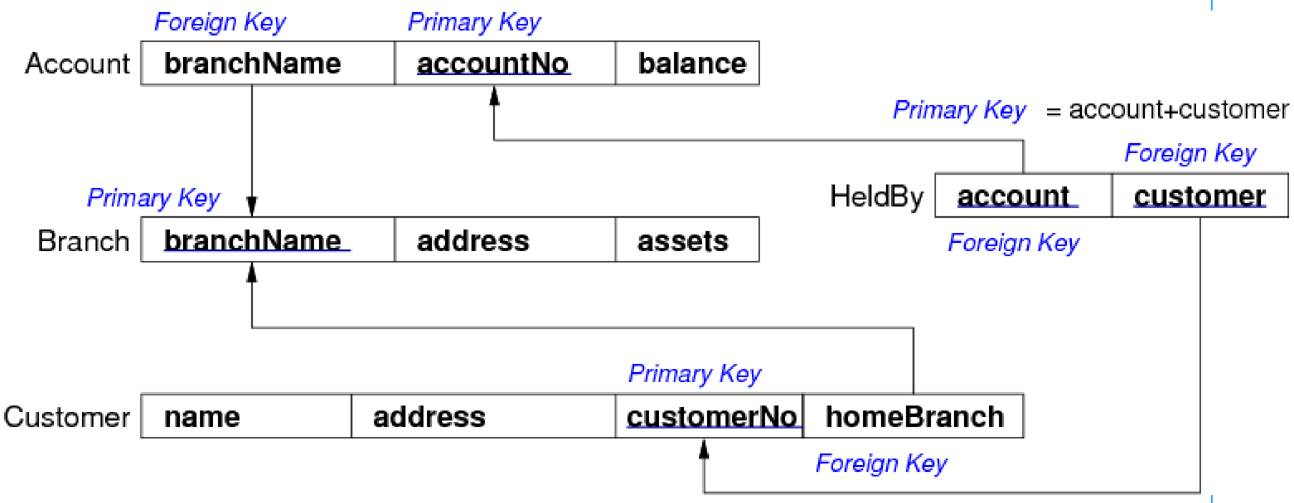
- `Class(...,Mon,2pm,Lyre,...)` is well-defined
- `Class(...,`**NULL**`,2pm,Lyre,...)` is not well-defined

❖ Referential Integrity

Referential integrity constraints

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

Example:



<< ^ >>

❖ Referential Integrity (cont)

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 $\{ R_1,\ R_2,\ \dots\ R_n \}$, and
- a set of integrity constraints

A relational database instance is

- a set of relation instances $\{ r_1(R_1),\ r_2(R_2),\ \dots\ r_n(R_n) \}$
- 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

Changes to the data fail if they would cause constraint violation

<< ∧

❖ Describing Relational Schemas

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

SQL provides a Data Definition Language (DDL) for this.

```
CREATE TABLE TableName (
   attrName_1 domain_1 constraints_1 ,
   attrName_2 domain_2 constraints_2 ,
   ...
   PRIMARY KEY (attr_i,attr_j,...),
   FOREIGN KEY (attr_x,attr_y,...)
               REFERENCES
               OtherTable (attr_m,attr_n,...), ...
);
```

To be continued ...

Produced: 10 Feb 2021