Relational Model

Dr. Odelu Vanga

Indian Institute of Information Technology Sri City

http://www.iiits.ac.in/people/regular-faculty/dr-odelu-vanga/

Basic Structure

- Formally, given sets $D_1, D_2, \ldots. D_n,$ a *relation* r is a subset of
  $D_1 \times D_2 \times \ldots \times D_n$
  That is, a relation is a set of n-tuples $(a_1, a_2, \ldots, a_n)$, where $a_i \in D_i$
- Example: if

  customer-name = {Jones, Smith, Curry, Lindsay}
  customer-street = {Main, North, Park}
  customer-city = {Harrison, Rye, Pittsfield}

Then, r = { (Jones, Main, Harrison),
(Smith, North, Rye),
(Curry, North, Rye),
(Lindsay, Park, Pittsfield)
}
is a relation over *customer-name* x *customer-street* x *customer-city*

customer-name	customer-street	customer-city
Jones	Main	Harrison
Smith	North	Rye
Curry	North	Rye
Lindsay	Park	Pittsfield

customer

Relation Schema

- Suppose $A_1, A_2, \ldots, A_n$ are *attributes*
- $R = (A_1, A_2, \ldots, A_n)$ is a *relation schema*

  E.g. *Customer-schema* = (*customer-name, customer-street, customer-city*)
- $r(R)$ is a *relation* on the *relation schema R*

  E.g. *customer (Customer-schema)*

attributes

customer-name	*customer-street*	customer-city
Jones	Main	Harrison
Smith	North	Rye
Curry	North	Rye
Lindsay	Park	Pittsfield

customer

Relation Instance

- The current values (*relation instance*) of a relation are specified by a table
- An element *t* of *r* is a *tuple*, represented by a *row* in a table

customer-name	*customer-street*	customer-city
Jones	Main	Harrison
Smith	North	Rye
Curry	North	Rye
Lindsay	Park	Pittsfield

tuples

customer

Relations are Unordered

- Order of tuples is irrelevant (tuples may be stored in an arbitrary order)
- E.g. *account* relation with unordered tuples

account-number	*branch-name*	*balance*
A-101	Downtown	500
A-215	Mianus	700
A-102	Perryridge	400
A-305	Round Hill	350
A-201	Brighton	900
A-222	Redwood	700
A-217	Brighton	750

Database

- A database consists of multiple relations
- Information about an enterprise is broken up into parts, with each relation storing one part of the information

E.g.: *account* : stores information about accounts
depositor : stores information about which customer owns which account
customer : stores information about customers

- Storing all information as a single relation such as
  bank(*account-number, balance, customer-name*, ..)
  results in
  - repetition of information (e.g. two customers own an account)

- **Normalization theory, deals with how to design relational schemas**

Banking Example

branch (*branch-name*, *branch-city*, *assets*)

Assume branch-name is unique

customer (*customer-name*, *customer-street*, *customer-city*)

Assume customer-name is unique

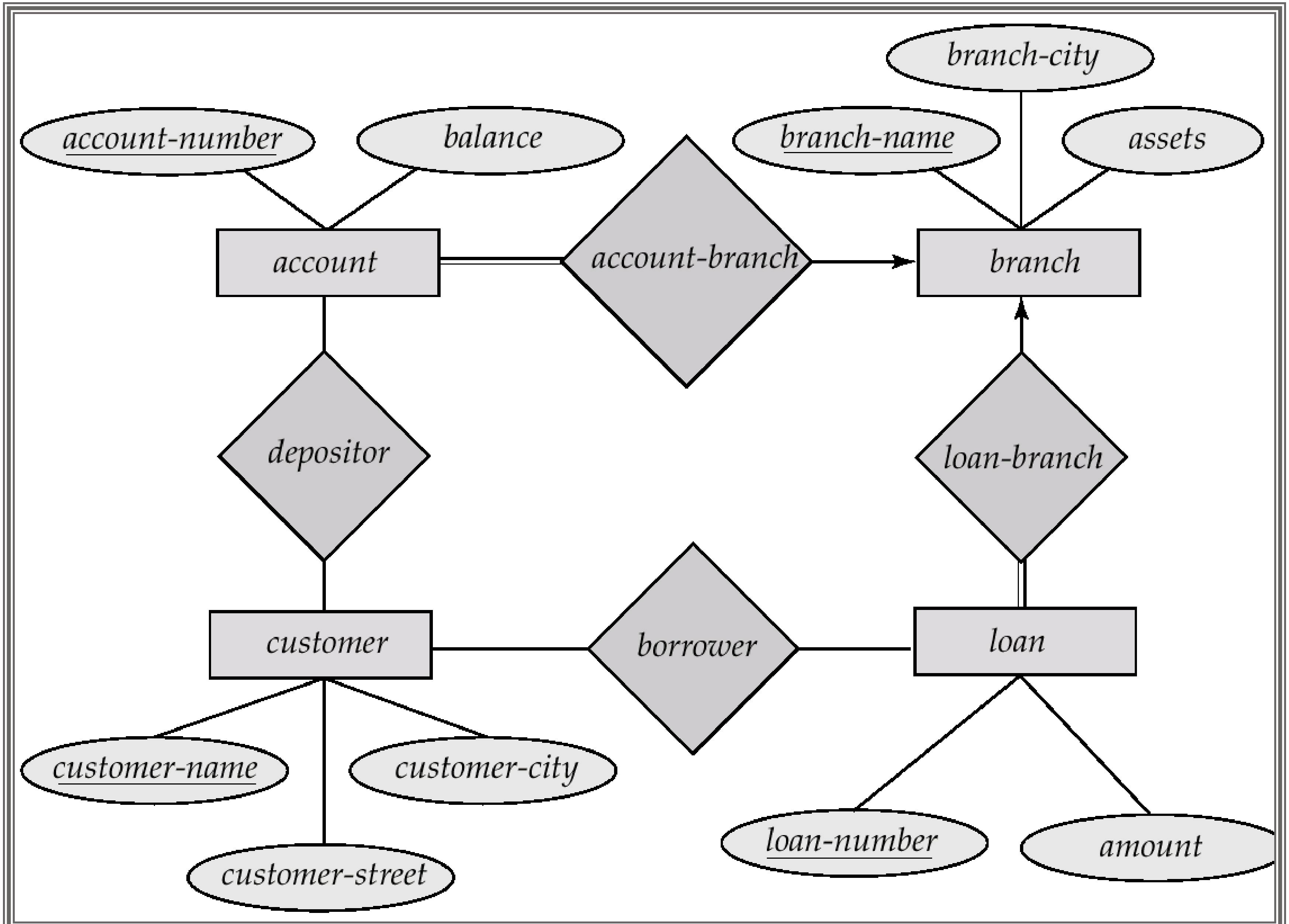
account (*account-number*, *branch-name*, *balance*)

loan (*loan-number*, *branch-name*, *amount*)

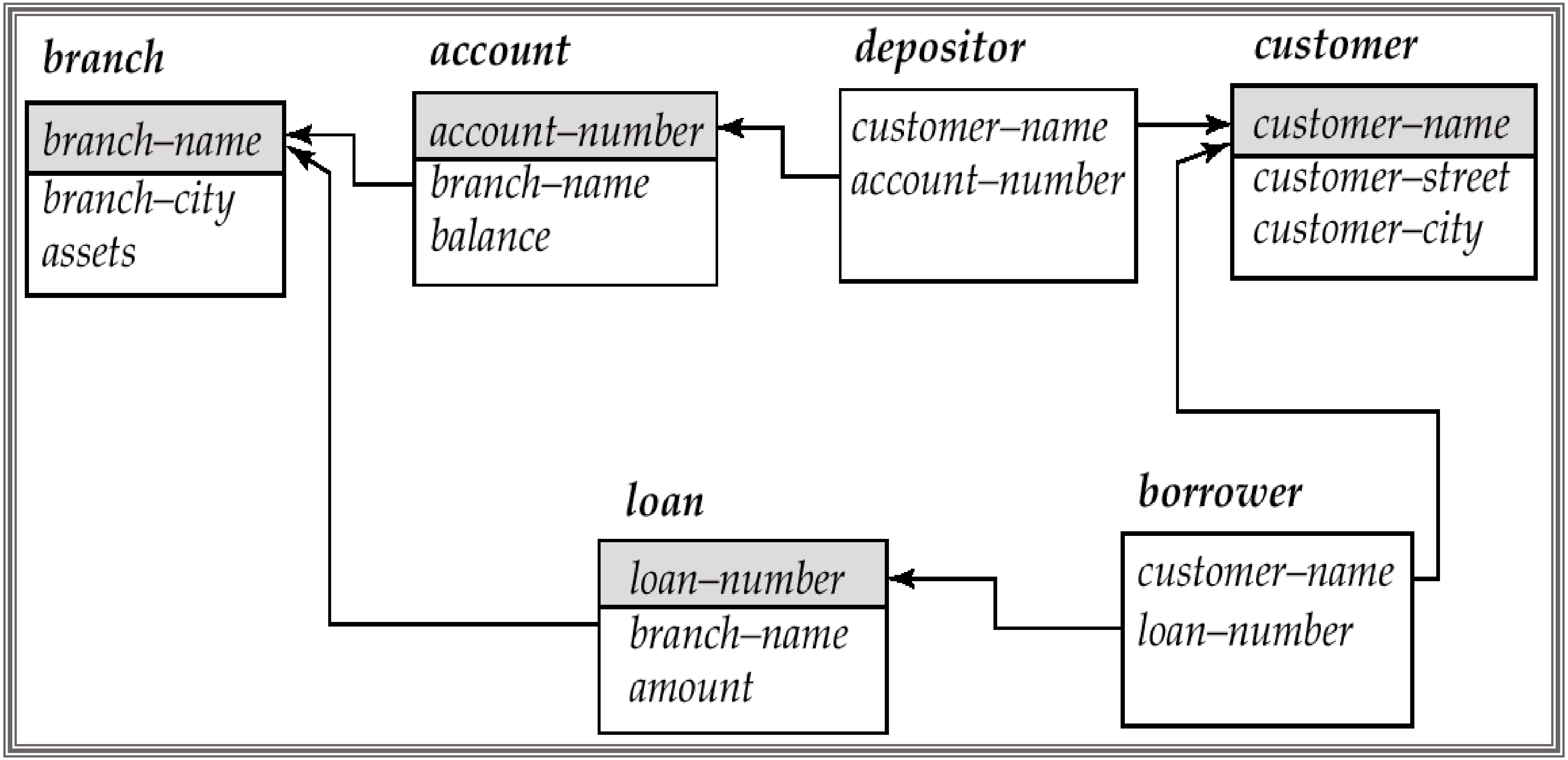
depositor (*customer-name*, *account-number*)

borrower (*customer-name*, *loan-number*)

E-R Diagram for the Banking Enterprise



Schema Diagram for the Banking Enterprise



Relational Algebra

- Procedural language
- Six basic operators
  - select
  - project
  - union
  - set difference
  - Cartesian product
  - rename
- **The operators take two or more relations as inputs and give a new relation as a result.**

Select Operation

- Notation: $\sigma_p(r)$
- p is called the selection predicate
- Defined as:

$$\sigma_p(\mathbf{r}) = \{t \mid t \in r \textbf{ and } p(t)\}$$

Where p is a formula in propositional calculus consisting of terms connected by : $\wedge$ (**and**), $\vee$ (**or**), $\neg$ (**not**)
Each term is one of:

<attribute> *op* <attribute> or <constant>

where *op* is one of: $=, \neq, >, \geq, <, \leq$

- Example of selection:

$$\sigma_{branch\text{-}name=\text{“}Perryridge\text{”}}(account)$$

Select Operation – Example

- Relation *r*

A	B	C	D
α	α	1	7
α	β	5	7
β	β	12	3
β	β	23	10

- $\sigma_{A \neq B \wedge C < D}(r)$

A	B	C	D
α	β	5	7

- $\sigma_{A=B \wedge D > 5}(r)$

A	B	C	D
α	α	1	7
β	β	23	10