Database Concept and Systems

CIT 314 (2 Units)

Bilkisu Muhammad-Bello (Mrs.)

Entity-Relationship Model

Basic Concepts continuation

Design of an E-R Database Schema

- The E-R data model gives us much flexibility in designing a database schema to model a given enterprise
- Among the designer's decisions are:
  - *Whether to use an attribute or an entity set to represent an object.*
  - *Whether a real-world concept is expressed more accurately by an entity set or* by a relationship set.
  - *Whether to use a ternary relationship or a pair of binary relationships.*
  - *Whether to use a strong or a weak entity set ; a strong entity set and its dependent weak entity sets may be regarded as a single "object" in the database, since weak entities are existence dependent on a strong entity.*
  - *Whether using generalization is appropriate; generalization, or a hierarchy of ISA relationships, contributes to modularity by allowing com*mon attributes of similar entity sets to be represented in one place in an E-R diagram.
  - *Whether using aggregation is appropriate; aggregation groups* a part of an E-R diagram into a single entity set, allowing us to treat the aggregate entity set as a single unit without concern for the details of its internal structure.

Design Phases

- The initial phase of database design, is to characterize fully the data needs of the prospective database users.
- The database designer needs to interact extensively with domain experts and users to carry out this task.
- The outcome of this phase is a **specification of user requirements**.
- Next, the designer chooses a data model, and by applying the concepts of the chosen data model, translates these requirements into a **conceptual schema** of the database.

Design Phases cont

- The schema developed at this conceptual-design phase provides a detailed overview of the enterprise.
- Stated in terms of the E-R model, the schema specifies all entity sets, relationship sets, attributes, and mapping constraints.
- The designer reviews the schema to confirm that all data requirements are indeed satisfied and are not in conflict with one another. She can also examine the design to remove any redundant features.
- Her focus at this point is describing the data and their relationships, rather than on specifying physical storage details

Design Phases cont

- A fully developed conceptual schema will also indicate the **functional requirements** of the enterprise.
- In a **specification of functional requirements**, users describe the kinds of operations (or transactions) that will be performed on the data. Example operations include modifying or updating data, searching for and retrieving specific data, and deleting data.
- At this stage of conceptual design, the designer can review the schema to ensure it meets functional requirements.

Design Phases cont

- The process of moving from an abstract data model to the implementation of the database proceeds in two final design phases.
  - ***Logical-design phase***, The designer maps the high-level conceptual schema onto the implementation data model of the database system that will be used.
  - ***Physical-design phase:*** The resulting system-specific database schema is used to specify physical features of the database such as the form of file organization and the internal storage structures.

Database Design for a Banking Enterprise

- **Bank Branches**: City, Unique name, Assets
- **Bank Customers**: Customer-id, Name, Street, City, account (loan), account officer/personal banker.
- **Bank Employees**: Employee-id, Name, Phone No, Dependants, Manager, Start date [length of employment]
- **Accounts**: [Savings and Current, Joint account, Multiple accounts], Unique account no, balance
- **Loans**: Originating branch, Loan number, Amount, Loan Payments, Date and amount for payments

Entity Sets Designation

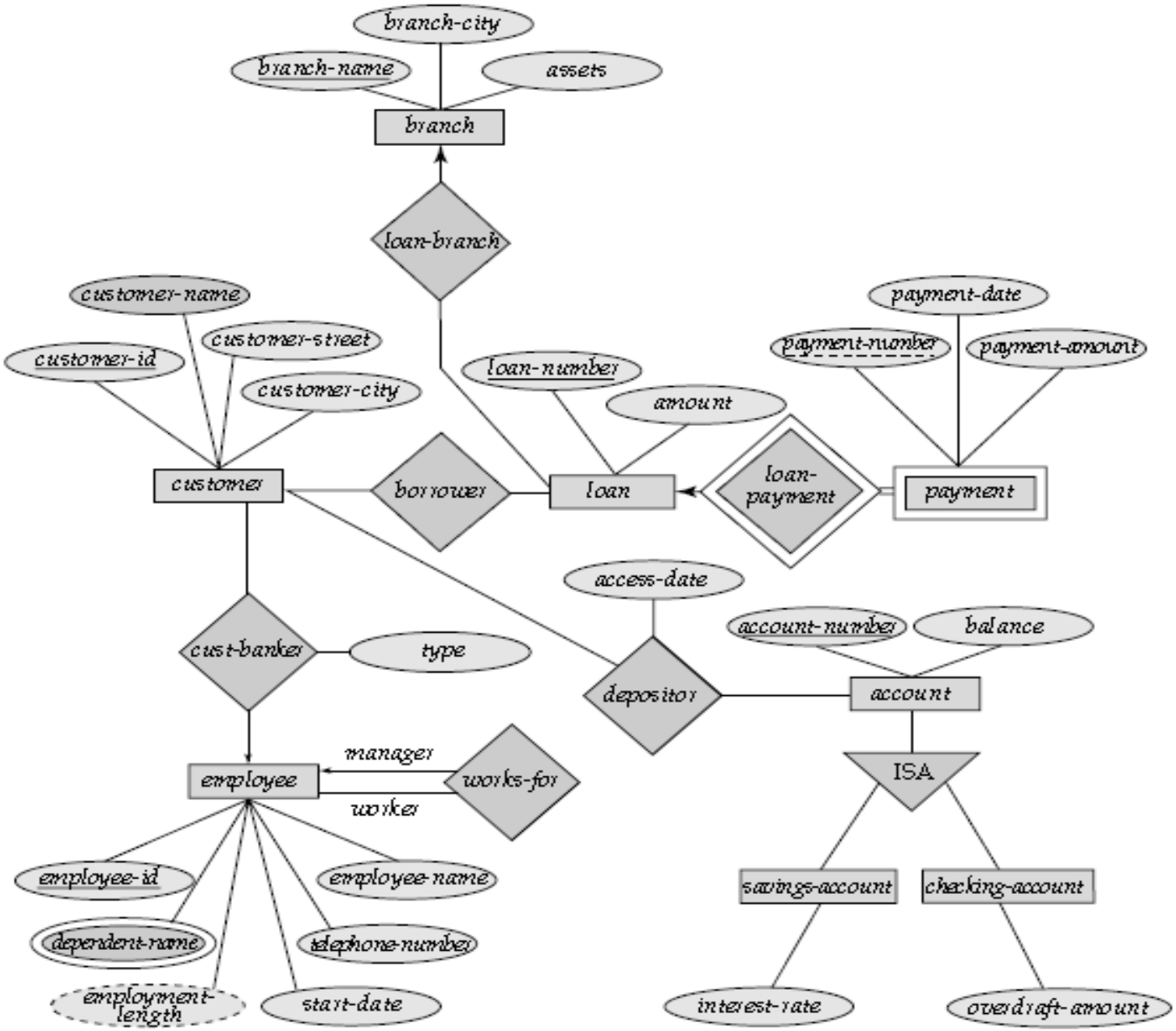
- ***Branch*** entity set, with attributes *branch-name, branch-city,* and *assets*.
- ***Customer*** entity set, with attributes *customer-id, customer-name, customer-street;* and *customer-city*. A possible additional attribute is *banker-name*.
- ***Employee*** entity set, with *attributes employee-id, employee-name, telephonenumber, salary,* and *manager*. Additional descriptive features are the multivalued attribute *dependent-name,* the base attribute *start-date,* and the derived attribute *employment-length*.

Entity Sets Designation cont

- Two account entity sets—***Savings-account*** and ***Checking-account***—with the common attributes of *account-number* and *balance*; in addition, savings-account has the attribute *interest-rate* and checking-account has the attribute *overdraft-amount.*
- ***Loan*** entity set, with the attributes *loan-number, amount*, and *originating-branch*.
- The weak entity set ***Loan-payment***, with attributes *payment-number, payment-date*, and *payment-amount*.

Relationship Sets Designation

- ***borrower***, a many-to-many relationship set between customer and loan.
- ***loan-branch***, a many-to-one relationship set that indicates in which branch a
- loan originated. Note that this relationship set replaces the attribute originating-branch of the entity set loan.
- ***loan-payment,*** a one-to-many relationship from loan to payment, which documents that a payment is made on a loan.
- ***depositor***, with relationship attribute access-date, a many-to-many relationship set between customer and account, indicating that a customer owns an account.
- ***cust-banker***,, a many-to-one relationship set expressing that a customer can be advised by a bank employee, and that a bank employee can advise one or more customers. Note that this relationship set has replaced the attribute banker-name of the entity set customer.
- ***works-for***, a relationship set between employee entities with role indicators manager and worker; the mapping cardinalities express that an employee works for only one manager and that a manager supervises one or more employees. Note that this relationship set has replaced the manager attribute of employee.


branch-city
branch-name
assets
branch
loan-branch
customer-name
customer-street
customer-id
customer-city
loan-number
amount
payment-date
payment-number
payment-amount
customer
borrower
loan
loan-payment
payment
access-date
cust-banker
type
depositor
account-number
balance
account
manager
employee
works-for
worker
ISA
employee-id
employee-name
dependent-name
telephone-number
employment-length
start-date
savings-account
checking-account
interest-rate
overdraft-amount

Reduction of an E-R Schema to Tables

- For each entity set and for each relationship set in the database, there is a unique table to which we assign the name of the corresponding entity set or relationship set.
- Both the E-R model and the relational-database model are abstract, logical representations of real-world enterprises, and both have similar design principles.
- Converting a database representation from an E-R diagram to a table format is the way we arrive at a relational-database design from an E-R diagram.

Tabular Representation of Strong Entity Sets

- Let *E* be a strong entity set with descriptive attributes *a1, a2, . . . , an*.
- We represent this entity by a table called *E* with *n* distinct columns, each of which corresponds to one of the attributes of *E*.
- Each row in this table corresponds to one entity of the entity set *E*.

Tabular Representation of Strong Entity Sets

loan-number	amount
L-11	900
L-14	1500
L-15	1500
L-16	1300
L-17	1000
L-23	2000
L-93	500

- Let D1 denote the set of all loan numbers
- Let D2 denote the set of all balances.
- Any row of the loan table must consist of a 2-tuple (v1, v2), where v1 is a loan (that is, v1 is in set D1) and v2 is an amount (that is, v2 is in set D2).
- In general, the loan table will contain only a subset of the set of all possible rows.
- We refer to the set of all possible rows of loan as the ***Cartesian product*** of D1 and D2, denoted by:
- $D1 \times D2$
- In general, if we have a table of n columns, we denote the Cartesian product of
- D1, D2, · · ·,Dn by
- $D1 \times D2 \times \cdots \times Dn-1 \times Dn$

Tabular Representation of Weak Entity Sets

- Let A be a weak entity set with attributes a1, a2, . . . , am.
- Let B be the strong entity set on which A depends.
- Let the primary key of B consist of attributes b1, b2, . . . , bn.
- We represent the entity set A by a table called A with one column for each attribute of the set:
- {a1, a2, . . . , am} ∪ {b1, b2, . . . , bn}

loan-number	*payment-number*	*payment-date*	*payment-amount*
L-11	53	7 June 2001	125
L-14	69	28 May 2001	500
L-15	22	23 May 2001	300
L-16	58	18 June 2001	135
L-17	5	10 May 2001	50
L-17	6	7 June 2001	50
L-17	7	17 June 2001	100
L-23	11	17 May 2001	75
L-93	103	3 June 2001	900
L-93	104	13 June 2001	200

Tabular Representation of Relationship Sets

- Let R be a relationship set
- Let a1, a2, . . . , am be the set of attributes formed by the union of the primary keys of each of the entity sets participating in R
- Let the descriptive attributes (if any) of R be b1, b2, . . . , bn. We represent this relationship set by a table called R with one column for each attribute of the set:
- {a1, a2, . . . , am} ∪ {b1, b2, . . . , bn}

customer-id	loan-number
019-28-3746	L-11
019-28-3746	L-23
244-66-8800	L-93
321-12-3123	L-17
335-57-7991	L-16
555-55-5555	L-14
677-89-9011	L-15
963-96-3963	L-17

Tools

- Many database systems provide tools for database design that support E-R diagrams.
- These tools help a designer create E-R diagrams, and they can automatically create corresponding tables in a database.
- There are also some database-independent data modeling tools that support E-R diagrams and UML class diagrams.
- These include
  - Rational Rose (www.rational.com/products/rose),
  - Visio Enterprise (see www.visio.com), and
  - ERwin (search for ERwin at the site www.cai.com/products).
  - Smart draw VP