The Entity-Relationship (E-R) Model

Designing a database is a form of real-world modelling: a query elicits an image of a real-world situation (e.g. loan status of a book)

The entity-relationship model of an enterprise
reflects its overall logical structure

entity = object that exists and is distinguishable from other objects

NB cf OOP object, where define instances of abstract classes:
doesn't exist until an instance is created

entity can be *abstract* or *concrete* (abstract ≠ class abstraction)
concrete person, book
abstract marriage, journey

entity set = set of entities of the same type
e.g. entity set *customer* = all persons having an account at bank

entity sets can intersect e.g. person / customer / employee

entity represented by a set of **attributes**
e.g. customer has customer_name, social_security, street, customer_city

each attribute has a set of permissible values defining its **domain**

particular customer
{(name, Brill), (ss#, 121-21-2121), (street, Putnam), (city, San Marcos)}

entity set <-> variable entity type <-> type definition

Example entity sets

branch b-name, b-city, assets
customer c-name, ss#, street, c-city
employee e-name, phone#
account account#, balance
transaction transaction#, date , amount

Relationships

relationship = association among several entities
relationship set (= mathematical relation on 2 or more entity sets)

Mathematical relation is defined as $\{(a,b,..., z) \mid a \in A, b \in B, ..., z \in Z\}$

binary if involves 2 entity sets
e.g. CustAcc =   has(customer, account)
                                  ... relationship set contains (Brill, 183)
... can have descriptive attributes associated with a relationship:
                                  e.g. last-access-date on a CustAcc relationship

ordering in a relationship may be significant cf WorksFor on employees.
                        => refer to role to distinguish when domain doesn't

There are no fixed rules about how to devise E-R model: choice of
relationship reflects real-world semantics

cf employee: e-name, phone#
     employee: e-name, phone   +   phone: phone#, location

Issues   does every employee have a phone?
             do employees have access to many phones?
             do employees share phones?

Note that can't play around with e-name as independent entity: have
        employee determines and is determined by e-name
        e-name is "not an independent entity"

No easy answer to what is entity / attribute set ...
                            depends on structure of the enterprise

Mapping cardinalities

1-1   every customer has exactly one account, no joint accounts
1-many   a customer can have > one accounts, no joint accounts
many-1   every customer has exactly one account, joint accounts
many-many a customer can have > one accounts, joint accounts

Existence dependencies

Can have an entity that exists only if another entity exists
                                    e.g. a transaction on an account

delete account => delete transactions from log but not v.v.

existence of x depends on existence of y =>
                            y **dominant** entity, x **subordinate** entity

Identity of entities: keys

"from a db perspective"
      difference between individual entities and relationships
                must be expressed in terms of their attributes

cf the possibility of observing objects in different states
              that happen to coincide *"from time to time"*

superkey = set of attributes that (taken together) identify an entity
e.g. ss# or {ss#,c-name} for a customer, but not c-name alone
              ... 2 people can have the same name

superkey without superfluous attributes is called a **candidate** key

candidate key = set of attributes that
          identifies an entity
          &
          is minimal wrt this property

normally select a key from the candidate keys as means to identify entity:
              call this the **primary** key

if an entity set has enough attributes to make a candidate key is
              a **strong entity set**
if an entity set doesn't have enough attributes to make a candidate key is
              a **weak entity set**
e.g. transaction#, date, amount may coincide on two separate accounts

weak entities are existence dependent on strong entity sets
              e.g. transaction on account
(where there is a weak entity set "there must be something around that
can distinguish conceptually different entities")

to distinguish entities in weak entity sets, identify first the strong entity set
on which it is existence dependent, and then find an attribute that will
discriminate between entities in the weak entity set if the strong entity is
known. E.g. (account#, transaction#) identifies transaction.

Keys for relationships

Keys distinguish entities in entity sets – what about relationships?

key for a **relationship** is derived from keys for its constituent entity sets
e.g. if R(A,B,...,Z) is a relationship set, then identify key "as a piece of info
about one or more of the entities participating in a relationship in R that is
sufficient to identify the relationship entirely". An appropriate way to
construct such a piece of info is to consider the union attribute(R) of the
sets of attributes

   primary_key(A), primary_key(B), ... , primary_key(Z), descAttr(R)

and pare down this set taking advantage of any functional dependency.

e.g. CustAcc   gives attribute(R) = {ss#, account#, date}

In this case, argue that
   if there is a many-1 relationship from customer to account
   then a candidate key is ss# (customer has at most one account)
   & the date associated with this account is uniquely spec by account#

Otherwise – if CustAcc many-many need both {ss#, account#} as key.

In general need to add such attributes from descAttr(R) as can't be inferred
through knowledge of functional dependency.
   e.g. CustBanker relationship may have a **role** field:
   loan officer / personal banker
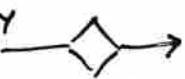   ... if *we can't infer* which role a banker plays for a customer
                                          must add *role* attribute to the key

Issues for object-based modelling

criterion	What is an entity? can a relationship be entity?
identity	how to tell the difference between entities? effect of evolution of entity on identity?
relationships	what connections are there between entities?
existence	dynamic vs static instantiation
dependencies	existence and functional
multiple views	what is whole of an entity? can we circumscribe its set of attributes
behaviour	do entities have state and behaviour?
time	do entities change, or just come and go?
hierarchy	inheritance, classification

AN ENTITY-RELATIONSHIP DIAGRAM

RELATIONSHIP SET — CustAcct

ENTITY SET — account

ATTRIBUTE — date
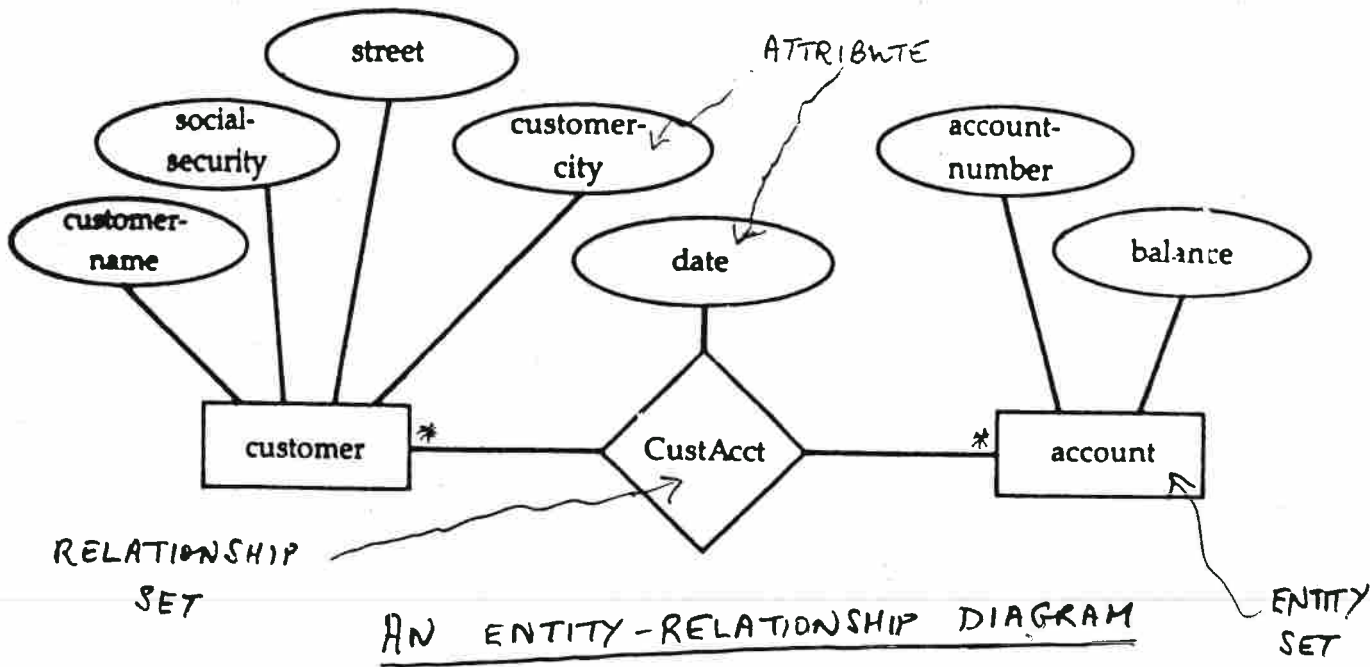
PUT ARROWS AT * TO INDICATE MAPPING CARDINALITY

MANY → ONE ETC.

DESCRIPTIVE ATTRIBUTE FOR RELATIONSHIP

CustAcct table

social-security	account-number	date
654-32-1098	259	17 June 1990
654-32-1098	630	17 May 1990
890-12-3456	401	23 May 1990
456-78-9012	700	28 May 1990
369-12-1518	199	13 June 1990
246-80-1214	467	7 June 1990
246-80-1214	115	7 June 1990
121-21-2121	183	13 June 1990
135-79-1357	118	17 June 1990
135-79-1357	225	19 June 1990
135-79-1357	210	27 June 1990

Customer table

customer-name	social-security	street	customer-city
Oliver	654-32-1098	Main	Harrison
Harris	890-12-3456	North	Rye
Marsh	456-78-9012	Main	Harrison
Pepper	369-12-1518	North	Rye
Ratliff	246-80-1214	Park	Pittsfield
Brill	121-21-2121	Putnam	Stamford
Evers	135-79-1357	Nassau	Princeton

Account table

account-number	balance
259	1000
630	2000
401	1500
700	1500
199	500
467	900
115	1200
183	1300
118	2000
225	2500
210	2200

account-number

PRIMARY
KEY
FOR
ACCOUNT

balance

transaction-
number

amount

date

log

account

transaction

DOMINANT
ENTITY

SUBORDINATE
ENTITY
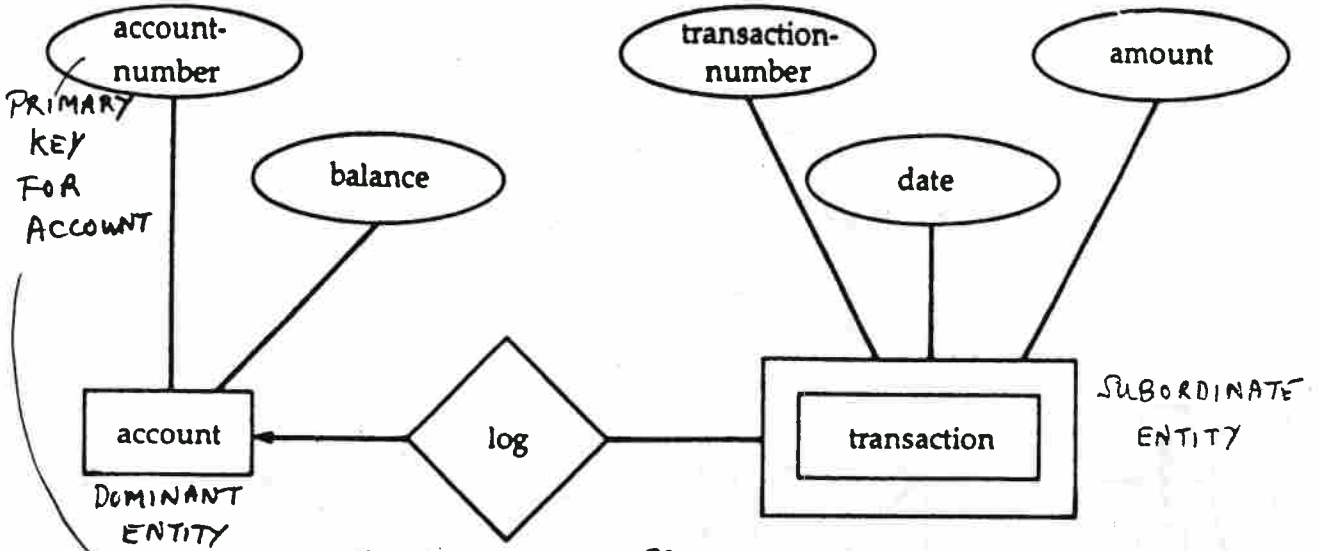
E-R DIAGRAM FOR WEAK ENTITY: TRANSACTION

TABLE
FOR
TRANSACTION

account-number	transaction-number	date	amount
259	5	11 May 1990	+50
630	11	17 May 1990	+70
401	22	23 May 1990	−300
700	69	28 May 1990	−500
199	103	3 June 1990	+900
259	6	7 June 1990	−44
115	53	7 June 1990	+120
199	104	13 June 1990	−200
259	7	17 June 1990	−79

GENERALISATION (c.f. OOP)

interest-
rate

savings-account

overdraft-
amount
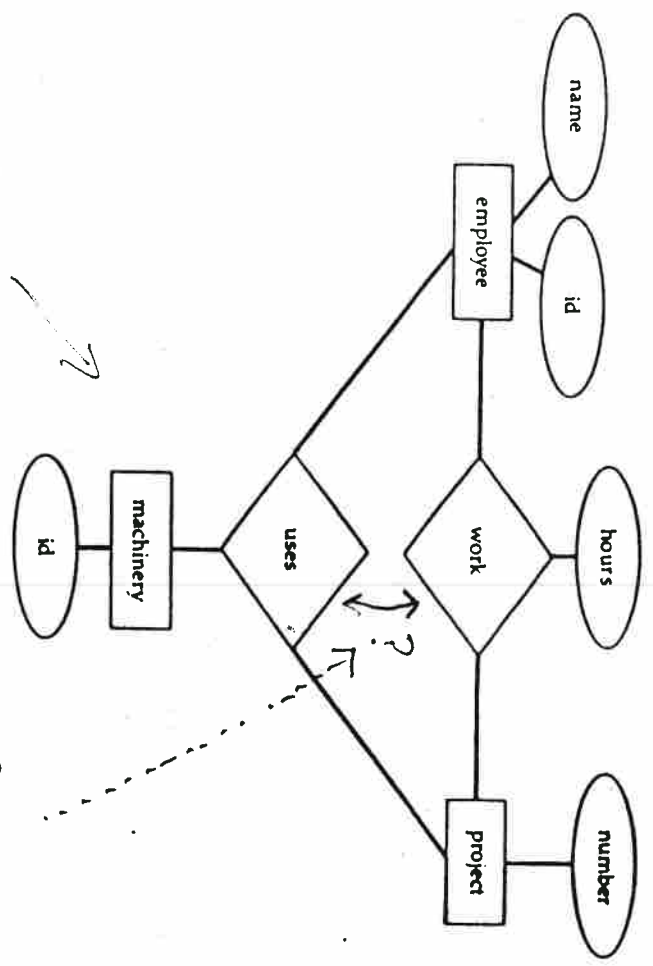
checking-account

ISA

account

account-
number

balance

TABULAR REPRESENTATIONS

1) ACCOUNT (ACCOUNT , S-ACCOUNT , C-ACCOUNT
ACCOUNT# BALANCE ACCOUNT# I-R ACCOUNT# O-A
2) S-ACCOUNT
ACCOUNT# BALANCE I-R
C-ACCOUNT
ACCOUNT# BALANCE O-A

OR

X
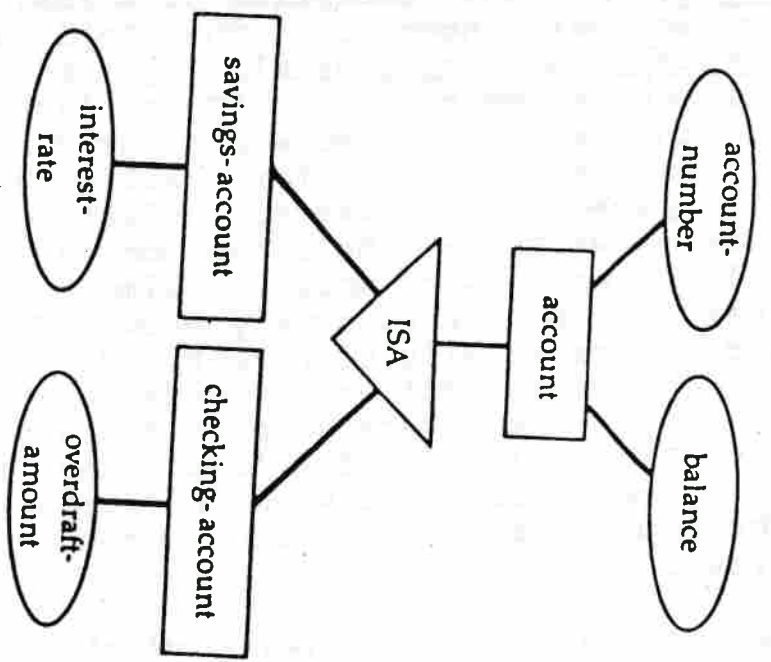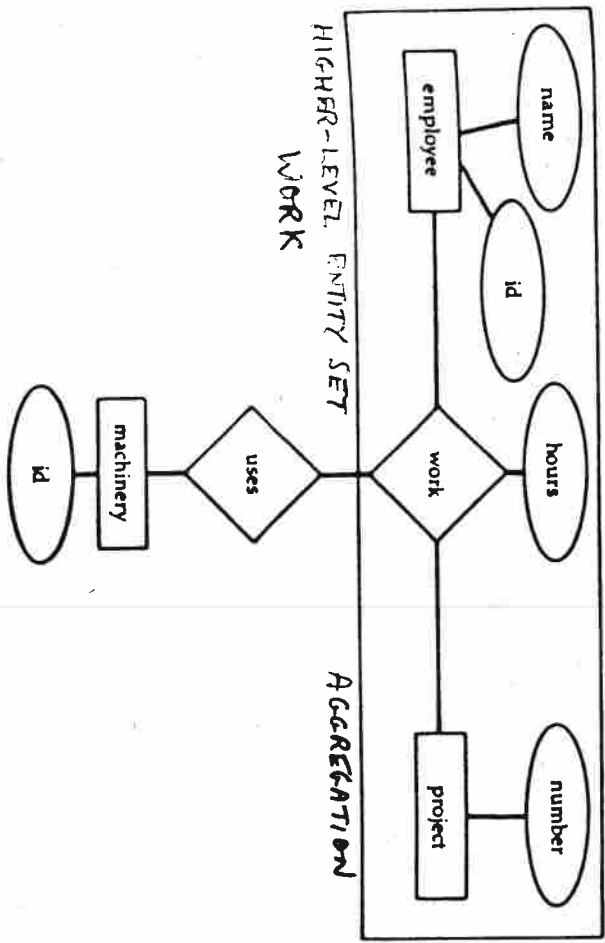
E-R DIAGRAM WITH REDUNDANT RELATIONSHIPS

"Hopi has <u>one noun that covers every thing or being that flies</u>, with the exception of birds, which class is denoted by another noun.... The Hopi actually call <u>insect</u>, <u>airplane</u>, and <u>aviator</u> all by the same word, and feel no difficulty about it.... This class seems to us too large and inclusive, but so would our class 'snow' to an Eskimo. We have the same word for falling snow, snow on the ground, snow packed hard like ice, slushy snow, wind-driven flying snow -- whatever the situation may be. To an Eskimo, this all-inclusive word would be almost unthinkable; he would say that falling snow, slushy snow, and so on, are sensuously and operationally different, different things to contend with; he uses different words for them and for other kinds of snow. The Aztecs go even farther than we in the opposite direction, with 'cold', 'ice', and 'snow' all represented by the same basic word with different terminations; 'ice' is the noun form; 'cold', the adjectival form; and for 'snow', 'ice mist'."

WHORF: LANGUAGE, THOUGHT & REALITY
1956

We are more ready to perceive things as entities when our language happens to have nouns for them. For what reason does our language happen to have the noun "schedule" for the connection between, say, a train and a time, but no such familiar noun for the connection between a person and his salary?

WILLIAM KENT: DATA & REALITY
1978