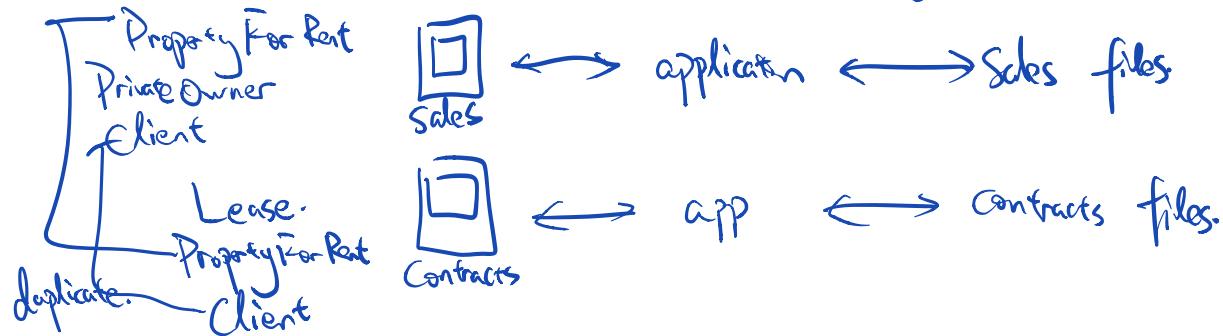


Compoorf DB.

Concept.

file-based systems.: ! systems that store data are not db systems

application perform services for users
each defines and manages own data.



Limitations

- Separation of data.
(usage of data held by other dept.).

- duplication of data.

 - same data held twice.

 - potential collisions / different formats.

 - data dependence. (on program)

 - file structure defined by program code.

causing ↓

- Incompatible file formats

- fixed queries. (new query → new program)

Database Approach.

Definition:

- a shared collection of logically related data.
- a self-describing collection of integrated records.

Data - entities

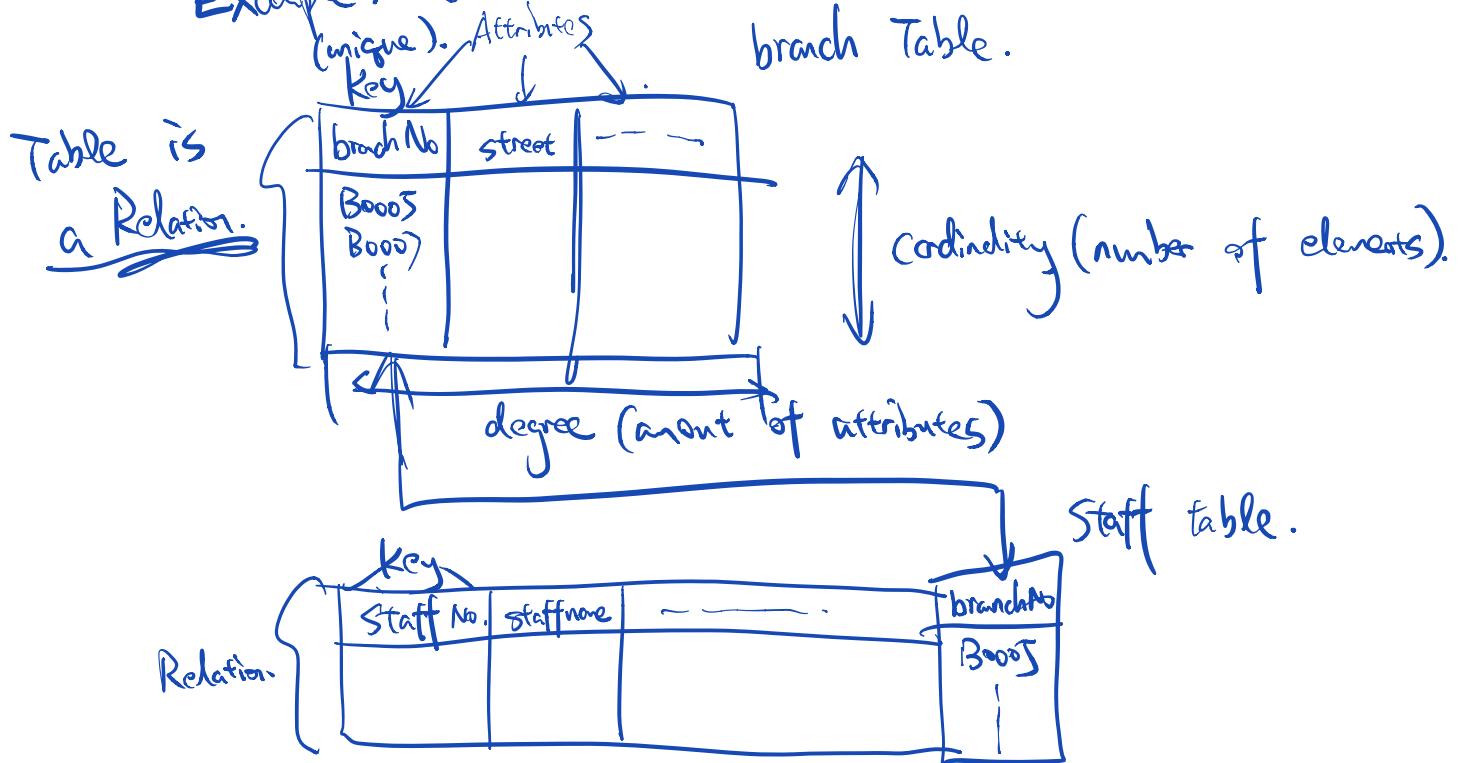
Attributes (describe entity)

Relationships.

(relational data model.)

data catalogue — descripm. data

Example. (relational data model)



Rows in table are unique defined by key

History.

first -gen
hierarchical & network.

Second -gen

Relational DB 1970s.

third -gen.

object relational
object-oriented

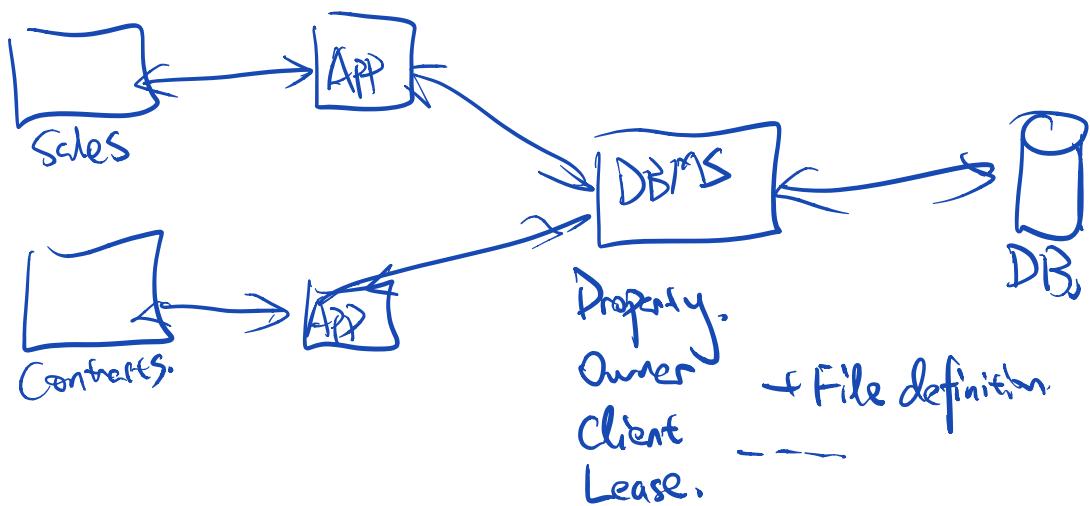
| DBMS (Management System)

| Definit.n.

System enables users
to define, create, maintain
a DB and provides
controlled access.

thru a lang. SQL.

provides services around DB.
integrity system.



Primary functions of a DBMS

- a data language Structured Query Lang SQL
- a security system
- a data integrity system.
- a Concurrency control system
- a recovery system
- a user accessible catalogue. (self-describing)

Advantages of DBMS.

- control of data redundancy
- data consistency
- more information from the same amount of data
(from relation)
- + integrity
- + security
- enforcement of data std.
- scale.
- resolve conflicts of requirements
- + concurrency.

- + accessibility & responsiveness

Disadvantages

- Complex
- Size
- conversion
- higher impact of failure.
- Cost
- additional hardware.
- performance.

SQL.

a data language. (key functions of a DBMS)

- Create and modify database structures.

Data Definition Language DDL

- manipulating data content of databases.

(changing and retrieving actual data)

Data Manipulation Language DML

Definition

SQL is a non-procedural and transform-oriented language. (Specify what information you require, rather how to get it.)

Essentially free form and semantically.

ISO Standard.

Create DB.

CREATE DATABASE Estate Agent;

USE Estate Agent

CREATE TABLE staff (StaffNo VARCHAR(4),
fName VARCHAR(20)) — salary DECIMAL(5,2),
—);

INSERT INTO 'staff' ('StaffNo', 'fName' —
— 'salary' —), VALUES ('SG17', 'Joe' —
— 21000, —)

DELETE FROM Staff WHERE StaffNo = 'SG17'.

DB can be populated from file.

Update data.

UPDATE Staff.

SET salary = salary * 1.03.

UPDATE Staff SET salary = salary * 0.95 WHERE
position = "Manager"

Underlying DB Engine.

InnoDB MyISAM.

↑
newer.

ACID-compliant transaction features
primary key - foreign key relationship
Support
Standard distribution.

Queries.

SELECT columns we want appear.

FROM table.

WHERE filter rows.

GROUP BY. forms groups of rows
 with same column val.

HAVING. filter groups subject

ORDER BY order results

SELECT * FROM Staff.

SELECT staffNo, fName, lName, salary FROM Staff.

SELECT DISTINCT propertyNo FROM Vierung;

SELECT Property - ----- (might get duplicates)

SELECT staffNo ----- salary/12 AS monthlySalary
From Staff
only label

not Variable

SELECT * FROM STAFF WHERE salary > 10000

Compound comparison search.

SELECT * FROM BRANCH, WHERE city = London OR city = Glasgow

SELECT ----- WHERE salary BETWEEN 20000
AND 30000.
or
salary ≥ 20000 AND
Salary ≤ 30000.

Set membership search. IN

SELECT --- FROM STAFF WHERE postin IN
("Manager", "Supervisor")

Pattern matching ; 'LIKE'.

SELECT propertyNo FROM --- WHERE
street LIKE '%some%'

% * any char seq.

= * any char once

LIKE '%Glasgow%'

LIKE 'G%'

LIKE 'G _____'

'IS NULL' search condition.

find stuff where there is no content.

SELECT Details FROM Viewings WHERE connect IS NULL.

IS NOT NULL.

SELECT * FROM Staff ORDER BY salary DESC.

ORDER BY type, rent DESC.

Aggregate functions.

COUNT SUM AVG MIN MAX.

SELECT COUNT(*) AS count FROM Property
WHERE rent > 350

SELECT COUNT(DISTINCT property No) AS count
FROM VIEWING

↑
distinct count of property
viewed.

SELECT COUNT(staffNo) AS count, SUM(salary)
AS sum

SELECT * FROM Staff
WHERE position = "Mngr"
Sum of managers' Salaries
SELECT MIN(salary) AS min FROM Staff.

'GROUP BY' querying sets of rows

SELECT branchNo, COUNT(staffNo) AS count,
SUM(salary) AS sum.
FROM staff GROUP BY branchNo.

SELECT may only contain

column names
aggregate func.
constant
expr. combined

— — —
FROM staff GROUP BY
branchNo HAVING COUNT(staffNo)
ORDER BY branch No

Subquery

SELECT embedded in a query.

SELECT * FROM Staff WHERE branchNo = (

SELECT branchNo
FROM Branch.

WHERE staffNo = '1'

variable set =

....").

SELECT * FROM Staff WHERE salary >

(SELECT AVG(salary) FROM Staff)

4 Subquery rules.

- ORDER BY should not be used
- Subquery only have one single col. (except exists)
- when used in comparison, should be on right hand side

ANY and ALL.

SOME.

↑
Conditioned true satisfied by ALL.
by at least one. value

SELECT * FROM Staff.

WHERE salary > SOME(SELECT salary
FROM Staff
where branchNo
= 'B003?')

EXISTS NOT EXIST.

SELECT * FROM Staff S.

WHERE EXISTS (SELECT * FROM Branch b.

WHERE s.branchNo
= b.branchNo
AND city = 'London'

Multi-Table Queries

use join if results come from other tables

FROM Staff Join Branch ON Staff.BranchNo
= Branch.BranchNo

Multiple grouping columns GROUP BY s.branchNo, s.staffNo

Inner Join.

SELECT b.* , p.*
FROM b, p
WHERE b.city = p.city
(excluding unmatched)

Left Outer Join.

SELECT --- FROM Branch AS b LEFT JOIN
Property AS p ON b.city = p.city
(have unmatched all branches)

Right Outer Join

would have unmatched Property

UNION.

produces result from queries and merge.

AUB

(SELECT city
from Branch) UNION (SELECT city
from Property)

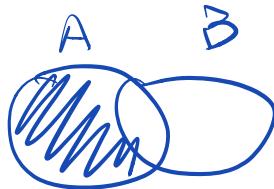
INTERSECT. (not supported by MySQL)

$A \cap B$

"Intersection"

INTERSECT

EXCEPT
 $A \setminus B$



DB Design.

ER diagram.

- Conceptual. (data modelling with entity relationship)
- Logical DB design.

Life cycle of DB.

sys. definition

db. planning

requirements gathering



Conceptual design.

ERD.

Logical design.

Database schema. (specified)

↓
Normalisation

↓
Database Schema. (verified)

DBMS Selection

Physical Design.

DB Schema Optimized.

Conceptual Design.

- understanding semantics of data.
- communication about info requirements.
- identify variation of user req.

Data model as entity relationship diagram. ERD, constituted of entities and relationships.

ER

* concepts

entities, relationships, and attributes

ER diagram

* directed graph.



* entity types / relationship types / attributes

ERD describe types of entity

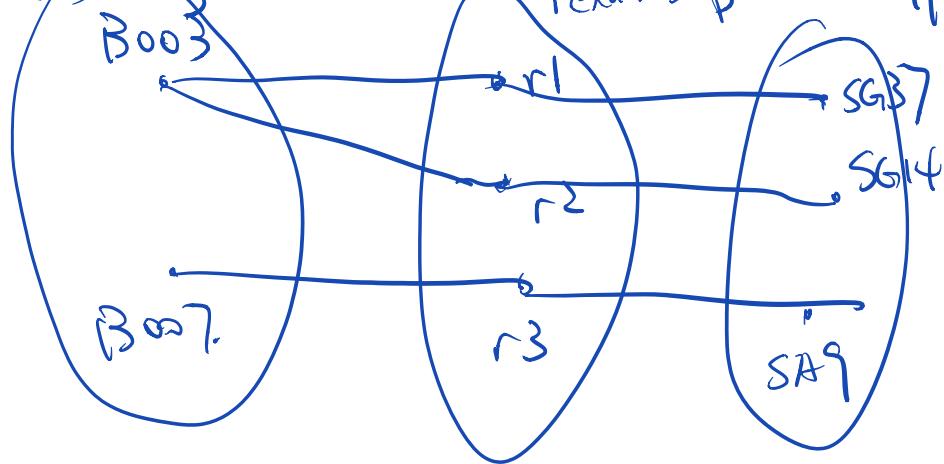
Entity physical or abstract.

Staff
not a person

Relationship Types

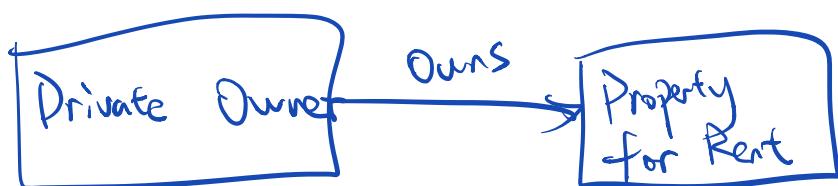
* set of meaningful associations among entity types.

Ex. Semantic net diagram
Branch entity relationship Staff entity

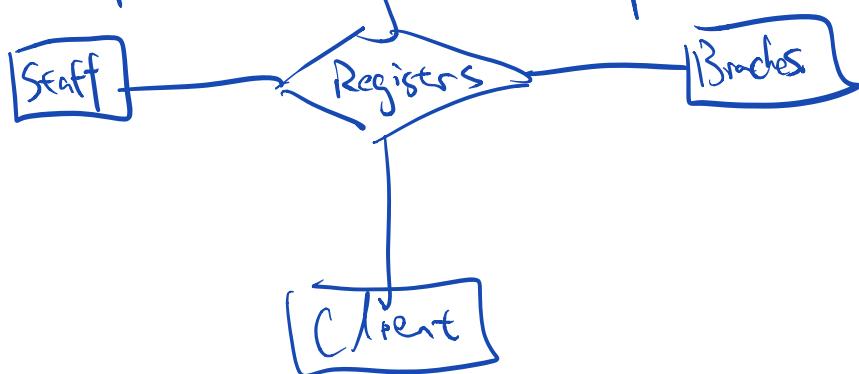


Degree of a relationship. : number of entities involved

binary
ternary
quaternary ---

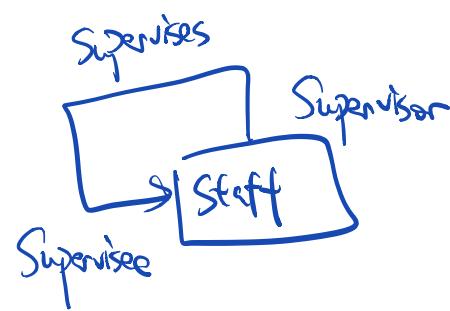


Example of a ternary relationship-

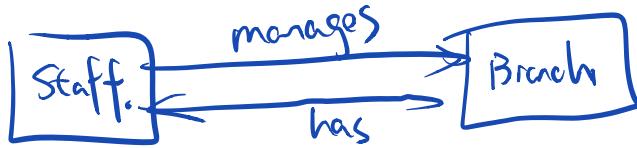


Recursive Relationship.

where same entity type participates more than once
in different roles.



Two entities may be associated than more than 1 relationship



Attributes.

- a property of an entity or a relationship type.
- hold values that describe an entity instance.

Domain

set of allowable values for one or more attributes

Int / String

Simple Attribute. / Atomic

attribute of single component

Composite Attribute.

Address multiple components

Single-valued Attribute.

holds a single value for each occurrence

Multi-valued Attribute. (multiple for each instance)

e.g. telephone numbers for an office.

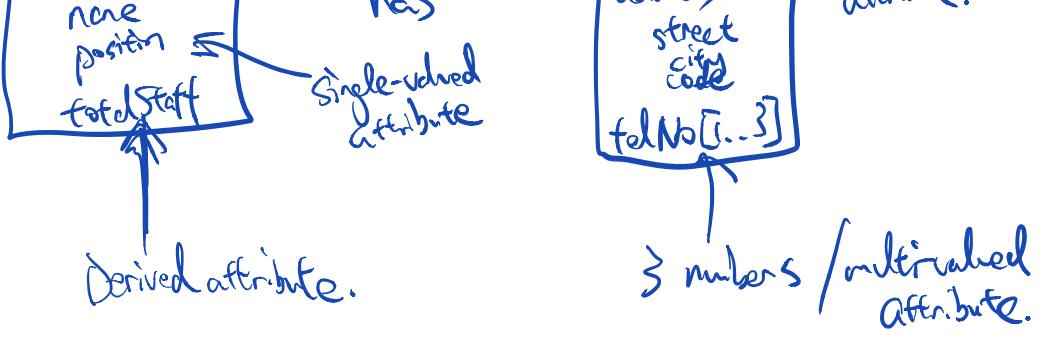
Derived Attribute.

Duration calculated from lease start to lease end

derivable.

Primary Key





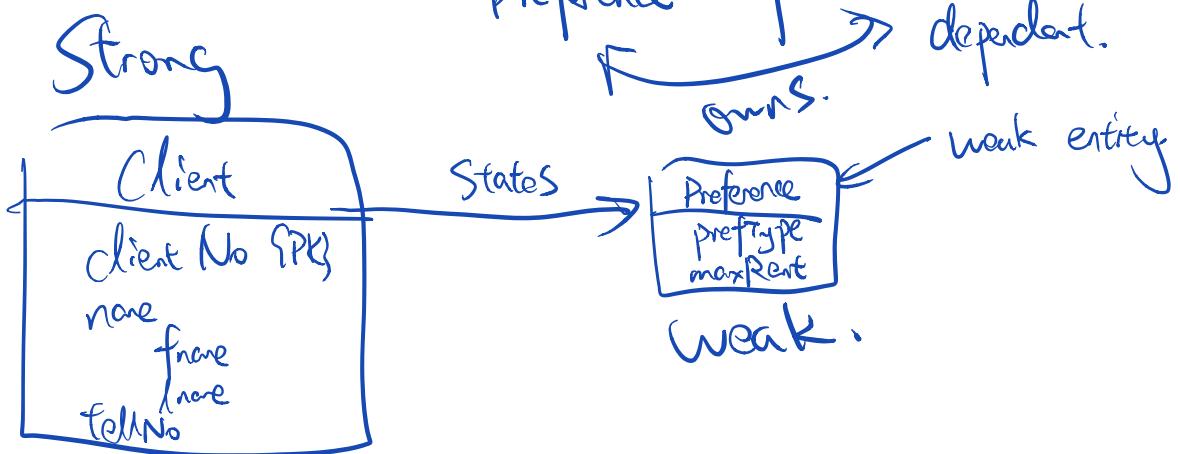
Entity Type

Strong — instance does not depend on other entity

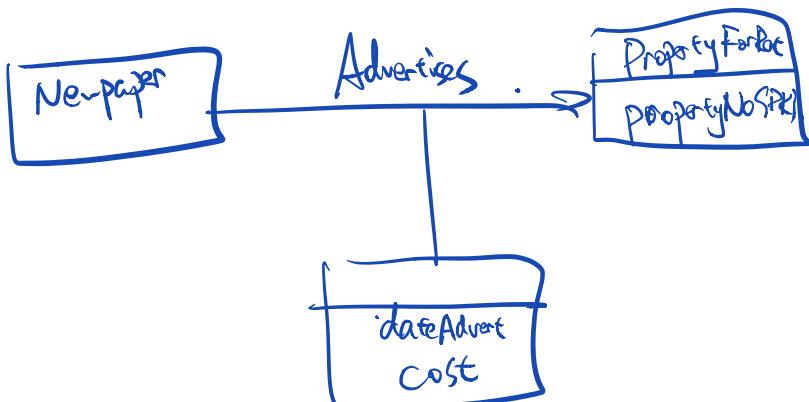
Weak — dependent on other entity

Preference of Client
owns.

dependent.



Relationship can have attributes



Structural Constraints.

Property must have an owner.

Branch must have Staff.

Main type.: multiplicity constraints.

number or range of possible occurrences
of an entity type + .

Represents policies established by company/user

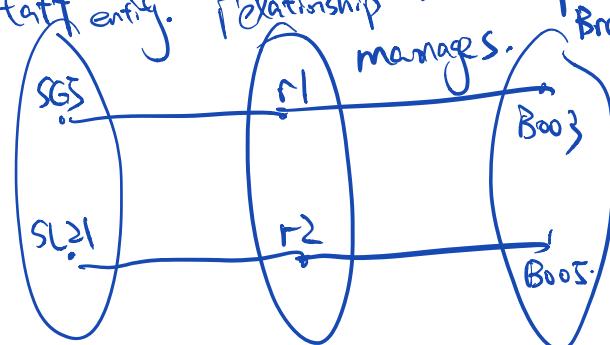
the most common degree of relationships is binary

one - to - one. 1:1

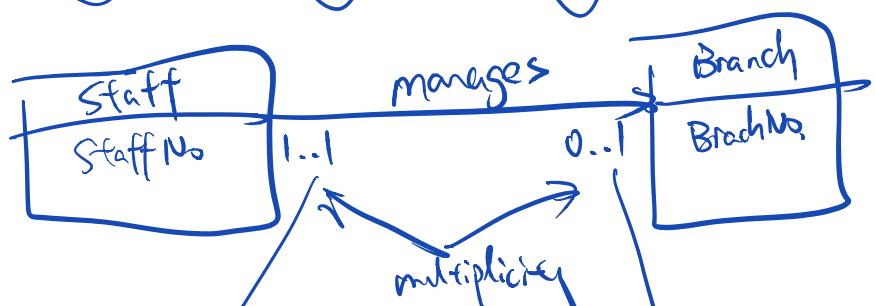
one - to - many 1: ∞

many - to - many ∞ : ∞

Semantic net example of 1:1 relationship
Staff entity. Relationship manages. Branch entity.



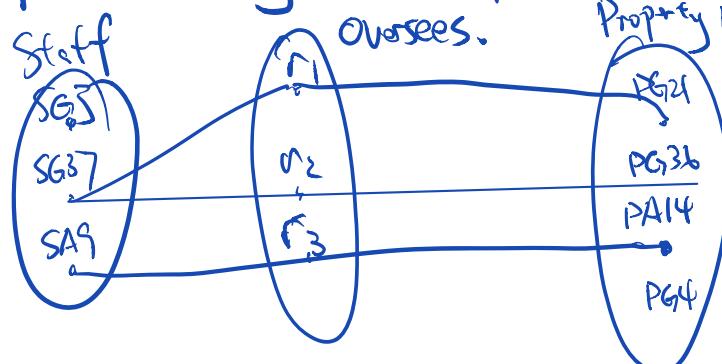
a Staff manages
at most one branch.



Each branch
managed by 1 staff

staff manages 0 or 1
branch.

Semantic net of 1 - many relationship

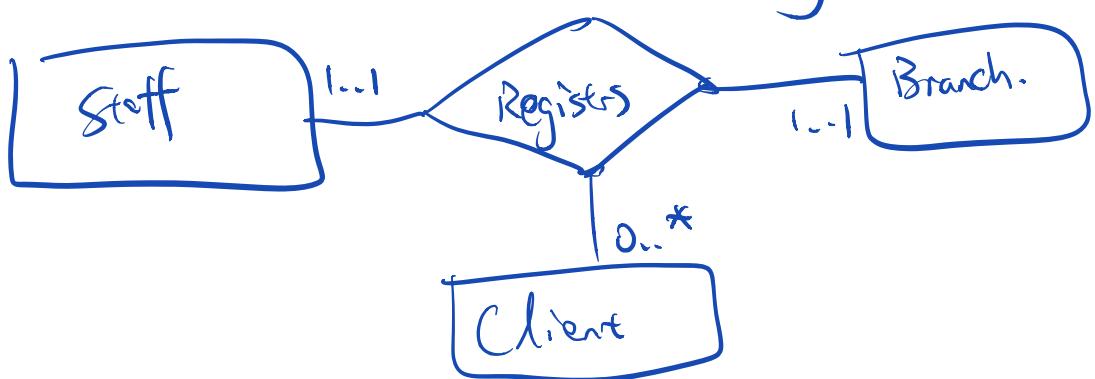
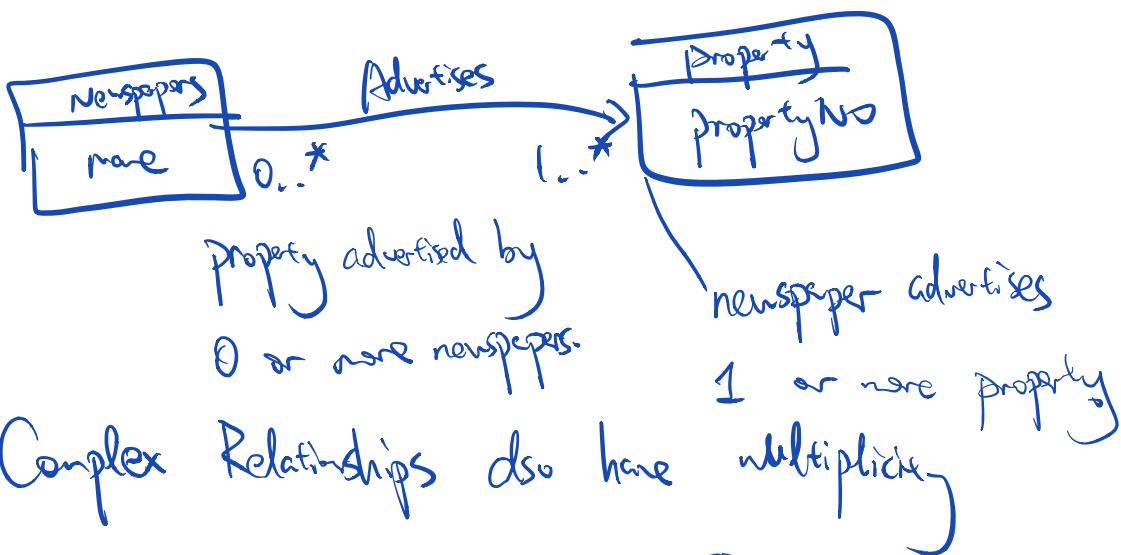
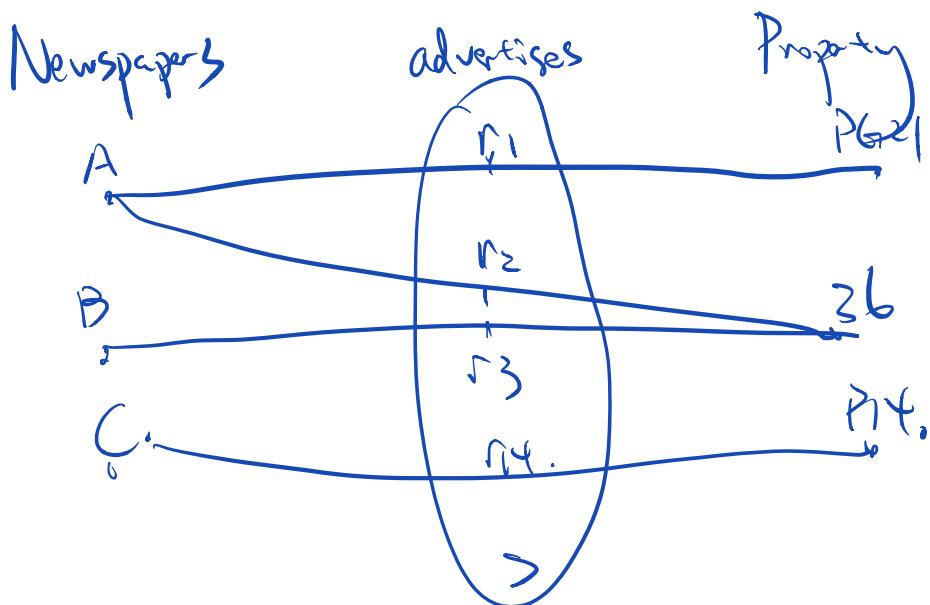


Oversees.

Property for Rent



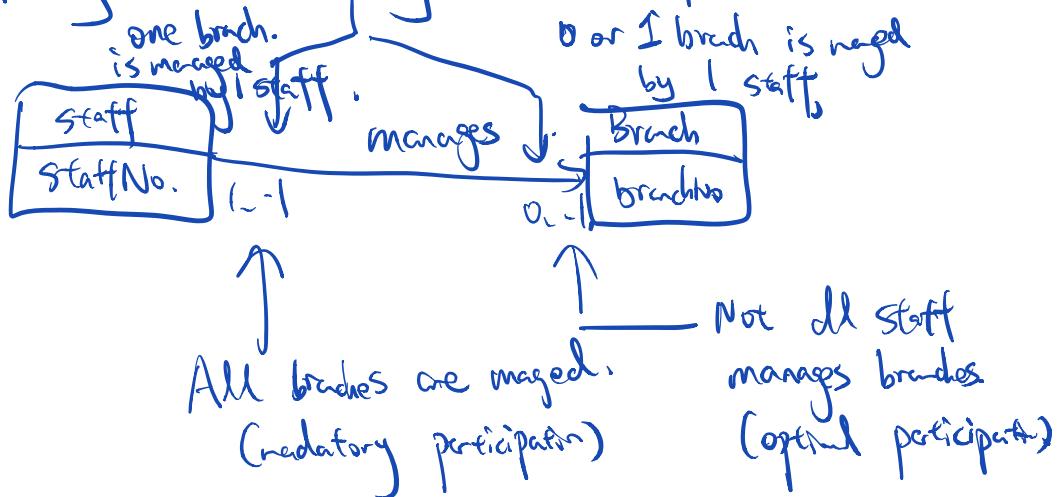
Specific net of many-many relationship.



a staff at a branch can register 0 or more clients.

a client at a branch will be req.
by only 1 member of staff

Multiplicity as Cardinality and Participation Constraints



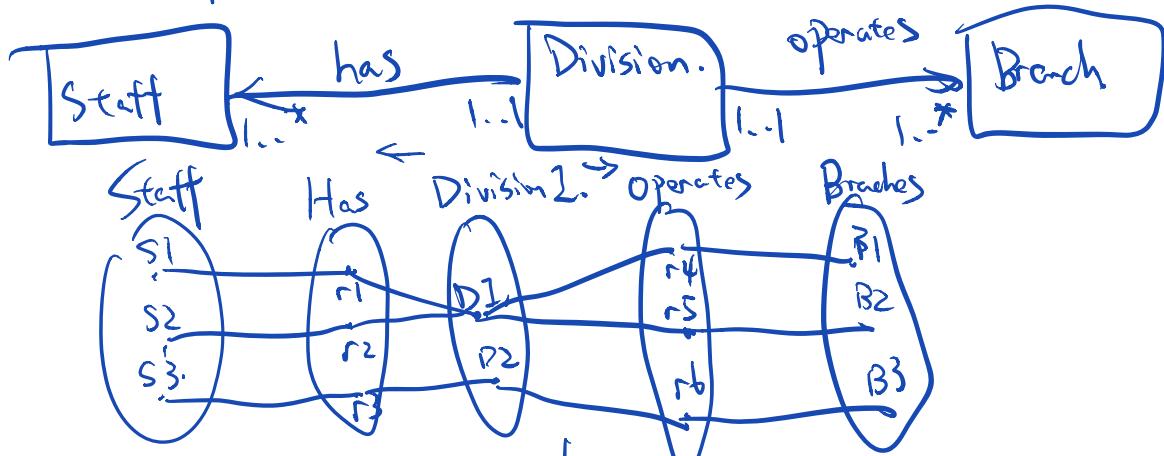
Multiplicity is the restriction

Cardinality - max number of possible relationship instances

Participation - whether some or all instance participate

Common traps:

Fan trap



Which branch S2 work for?

Restructure this:



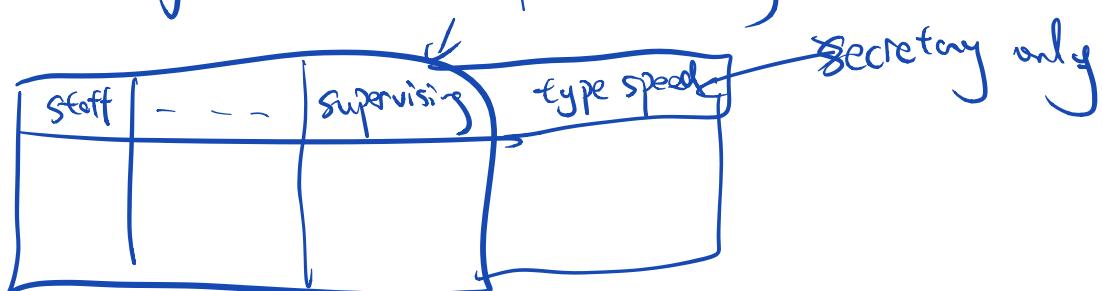
Chasm Trap (Skipped)

Enhanced Entity Relationship Model.

Addition of EER.

- * Specialisation / generalisation
- * aggregation
- * composition

Specialisation/ generalisation. supervisor only



* superclass Staff

* subclass Secretary

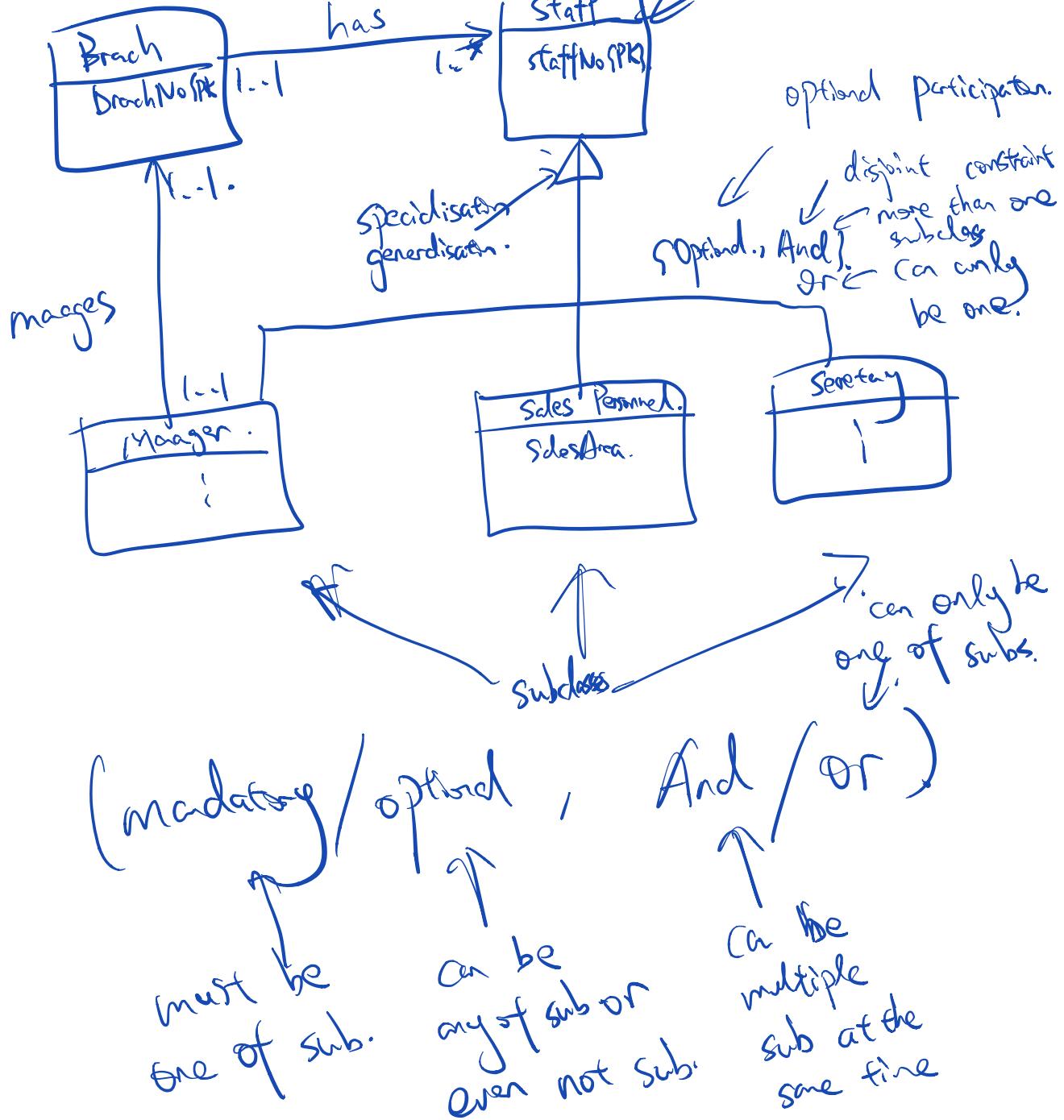
Superclass/subclass 1:1 relationship.

Superclass may contain overlapping / distinct subclasses

not all members of Superclass has to be a member of subclasses.

- Attribute Inheritance

Super class.



Logical Design.

- * objective of a data model

create an accurate representation of data relation

- * logical design uses the data model to specify data structures.

- define tables

- translates ER into schema

Strong entity types

don't rely on existence of another entity

- create a table for each strong entity
- include simple attributes
- for composite attributes

Client(clientNo, fname, ...)

Weak entity types

rely on existence of another entity. (e.g.

Preference rely on existence of Client)

- create a table for each weak entity
- include all simple attributes
- pk partially or fully derived from parent

Preference (clientNo, prefType, maxRent)

I. Binary Relationships