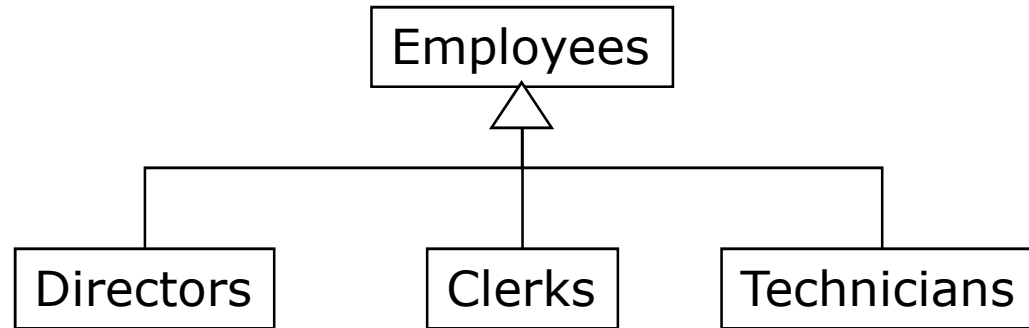


Generalization/Specialization (I)



Employees (emp, generic attributes)

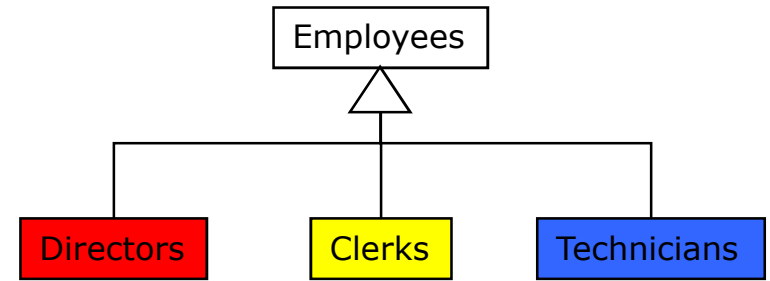
Technicians (emp, specific attributes)

Directors (emp, specific attributes)

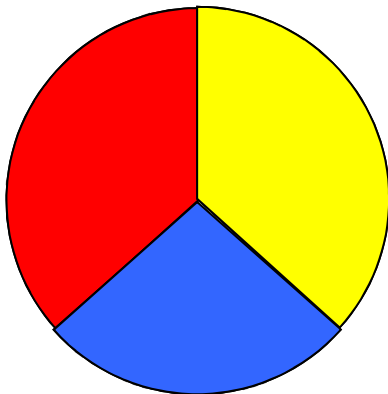
Clerks (emp, specific attributes)

Generalization/Specialization (II)

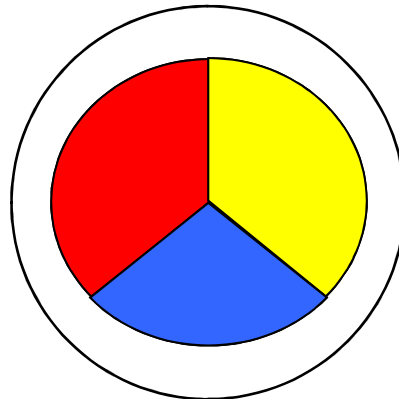
- Complete/Incomplete
- Disjoint/Overlapping



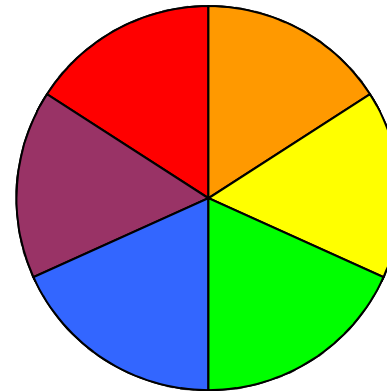
Complete Disjoint



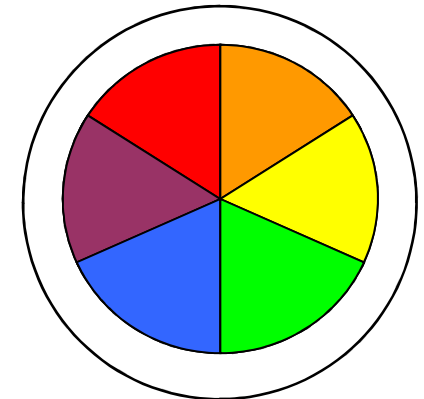
Incomplete Disjoint



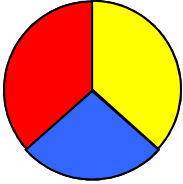
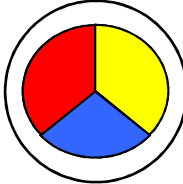
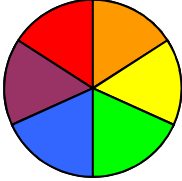
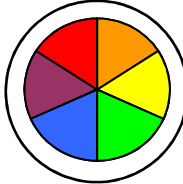
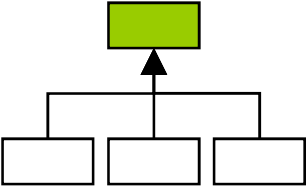
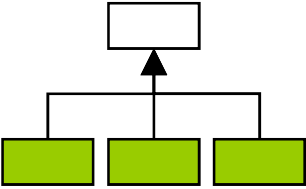
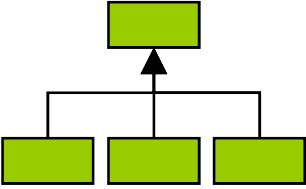
Complete Overlapping



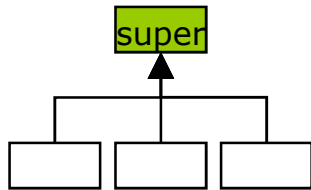
Incomplete Overlapping



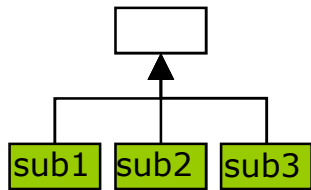
Generalization/Specialization (III)

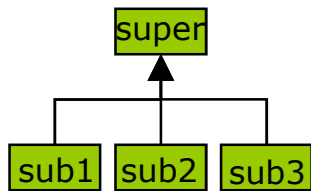
Generalization/Specialization (IV)



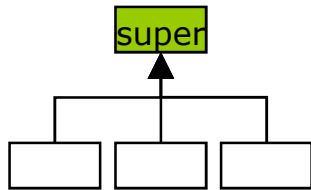
SELECT *
FROM super;



SELECT *
FROM sub1
UNION
SELECT *
FROM sub2
UNION
SELECT *
FROM sub3;

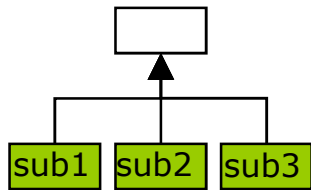


Generalization/Specialization (IV)



SELECT *
FROM super;

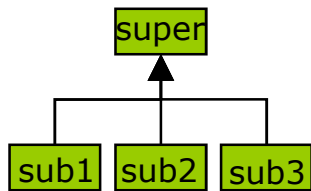
sub1(<u>a</u> ,	f)
	1	f1
	2	f2
	3	f3



SELECT *
FROM sub1
UNION
SELECT *
FROM sub2
UNION
SELECT *
FROM sub3;

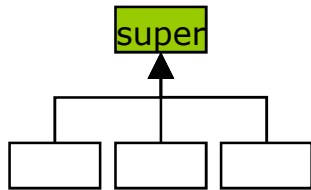
super(<u>a</u> ,	b,	c,	d,	e)
	1	b1	c1	d1	e1
	2	b2	c2	d2	e2
	3	b3	c3	d3	e3

sub2(<u>a</u> ,	g)
	1	g1



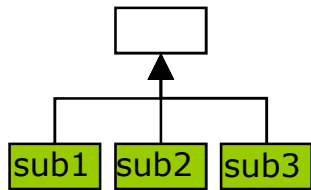
sub3(<u>a</u> ,	h)
	1	h1
	2	h2

Generalization/Specialization (IV)



SELECT *
FROM super;

sub1(<u>a</u> ,	f)
	1	f1
	2	f2
	3	f3

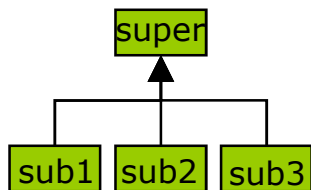


SELECT *
FROM sub1
UNION
SELECT *
FROM sub2
UNION
SELECT *
FROM sub3;

super(<u>a</u> ,	b,	c,	d,	e)
	1	b1	c1	d1	e1
	2	b2	c2	d2	e2
	3	b3	c3	d3	e3

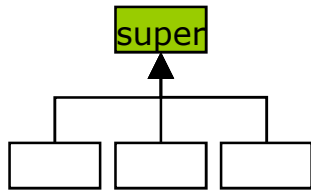
sub2(<u>a</u> ,	g)
	1	g1

sub3(<u>a</u> ,	h)
	1	h1
	2	h2



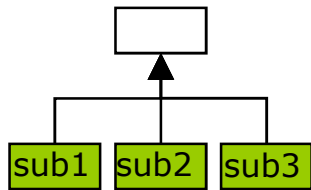
SELECT *
FROM super p, sub1 b1, sub2 b2, sub3 b3
WHERE p.a=b1.a AND p.a=b2.a AND p.a=b3.a;

Generalization/Specialization (IV)



SELECT *
FROM super;

sub1(<u>a</u> ,	f)
1	f1	
2	f2	
3	f3	

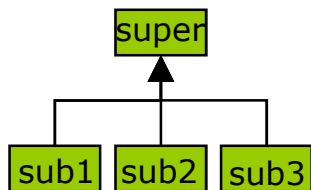


SELECT *
FROM sub1
UNION
SELECT *
FROM sub2
UNION
SELECT *
FROM sub3;

super(<u>a</u> ,	b,	c,	d,	e)
1	b1	c1	d1	e1	
2	b2	c2	d2	e2	
3	b3	c3	d3	e3	

sub2(<u>a</u> ,	g)
1	g1	

sub3(<u>a</u> ,	h)
1	h1	
2	h2	



~~SELECT *
FROM super p, sub1 b1, sub2 b2, sub3 b3
WHERE p.a=b1.a AND p.a=b2.a AND p.a=b3.a;~~

(a,	b,	c,	d,	e,	a,	f,	a,	g,	a,	h)
1	b1	c1	d1	e1	1	f1	1	g1	1	h1	

Generalization/Specialization (V)

R(a, b)
1 a
2 b
3 ?

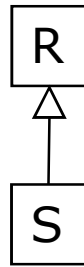
S(b, c)
a 4
c 5

R LeftOuterJoin S(a, b, b', c)
1 a a 4
2 b ? ?
3 ? ? ?

R RightOuterJoin S(a, b, b', c)
1 a a 4
? ? c 5

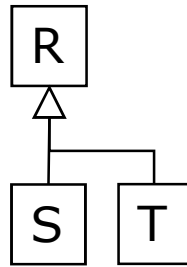
R FullOuterJoin S(a, b, b', c)
1 a a 4
2 b ? ?
3 ? ? ?
? ? c 5

Generalization/Specialization (VI)



```
SELECT R.a, R.b, S.c
FROM R, S
WHERE R.b=S.b
UNION
SELECT R.a, R.b, NULL
FROM R
WHERE R.b NOT IN ( SELECT S.b
                   FROM S);
```

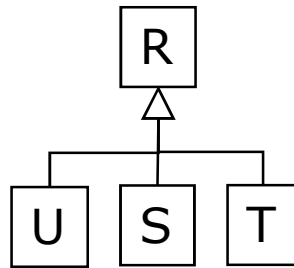
Generalization/Specialization (VI)



```
SELECT R.a, R.b, S.c
FROM R, S
WHERE R.b=S.b
UNION
SELECT R.a, R.b, NULL
FROM R
WHERE R.b NOT IN ( SELECT S.b
                   FROM S);
```

How many “UNION” are needed with 2 subclasses?

Generalization/Specialization (VI)



```

SELECT R.a, R.b, S.c
FROM R, S
WHERE R.b=S.b
UNION
SELECT R.a, R.b, NULL
FROM R
WHERE R.b NOT IN ( SELECT S.b
                   FROM S);
  
```

How many “UNION” are needed with 2 subclasses?
And with 3?

Outer Join in SQL'99 (I)

<table1> [CROSS | INNER | [LEFT|RIGHT|FULL] OUTER] JOIN <table2>
[ON <condition>]

- The order in the FROM clause is not commutative now
 - Joins are performed from left to right
- Predicate is evaluated after the outer join

Outer Join in SQL'99 (II)

Left

```
SELECT p.a, b, c, d, e, f, g, h
FROM super p LEFT OUTER JOIN sub1 b1 ON p.a=b1.a
      LEFT OUTER JOIN sub2 b2 ON p.a=b2.a
      LEFT OUTER JOIN sub3 b3 ON p.a=b3.a;
```

(<u>a</u> ,	b,	c,	d,	e,	f,	g,	h)
1	b1	c1	d1	e1	f1	g1	h1	
2	b2	c2	d2	e2	f2	?	h2	
3	b3	c3	d3	e3	f3	?	?	

Right

```
SELECT p.a, b, c, d, e, f, g, h
FROM super p RIGHT OUTER JOIN sub1 b1 ON p.a=b1.a
      RIGHT OUTER JOIN sub2 b2 ON p.a=b2.a
      RIGHT OUTER JOIN sub3 b3 ON p.a=b3.a;
```

Full

```
SELECT p.a, b, c, d, e, f, g, h
FROM super p FULL OUTER JOIN sub1 b1 ON p.a=b1.a
      FULL OUTER JOIN sub2 b2 ON p.a=b2.a
      FULL OUTER JOIN sub3 b3 ON p.a=b3.a;
```