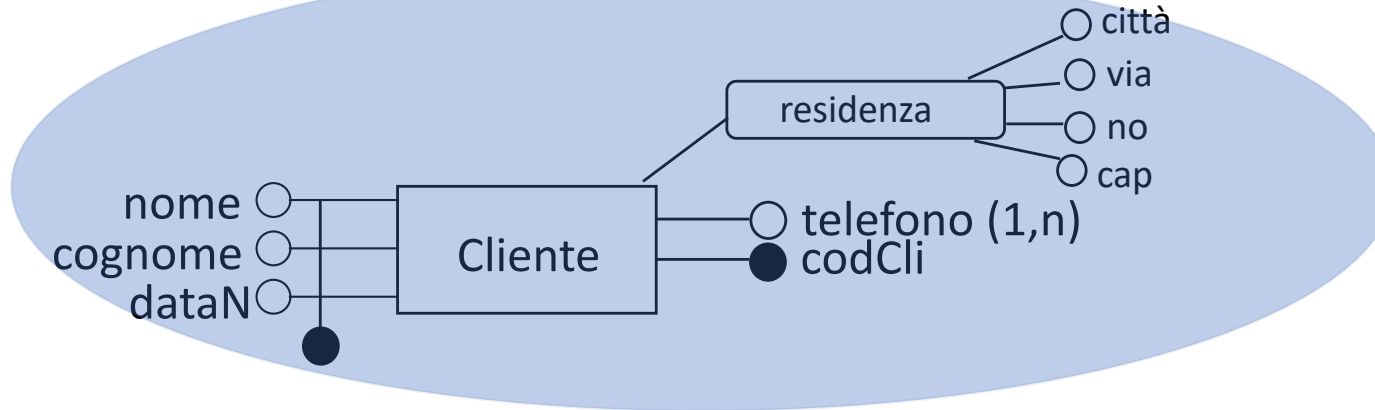


Tuning logico,  
denormalizzazione e  
modello nested relational

# ristrutturazione e traduzione



CLIENTE(codCli, nome, cognome, dataN, città, via, no, cap)

RISPONDEA(codCli<sup>CLIENTE</sup>, telefono)



schema in forma normale (no ridondanza)



determinare nome, cognome e numeri di telefono dei clienti (o di un cliente) è costoso: richiede un **join**

SELECT nome, cognome, telefono

FROM CLIENTE NATURAL JOIN RISPONDEA

# possibile denormalizzazione

CLIENTE(codCli,*nome*,*cognome*,*dataN*,città,via,no,*cap*,telefono)

codCli	nome	cognome	dataN	città	via	no	cap	telefono
6610	anna	rossi	05/10/1979	genova	via scribanti	16	16131	01055664433
6610	anna	rossi	05/10/1979	genova	via scribanti	16	16131	34012345678
6635	paola	bianchi	12/04/1976	genova	via dodecaneso	35	16146	0104647992
6635	paola	bianchi	12/04/1976	genova	via dodecaneso	35	16146	0103536699
6635	paola	bianchi	12/04/1976	genova	via dodecaneso	35	16146	34812345678



ridondanza:

nome, cognome, ... sono ripetuti per ogni numero di telefono

# possibile denormalizzazione

CLIENTE(codCli,*nome*,*cognome*,*dataN*,*città*,*via*,*no*,*cap*,telefono)

codCli	nome	cognome	dataN	città	via	no	cap	telefono
6610	anna	rossi	05/10/1979	genova	via scribanti	16	16131	01055664433
6610	anna	rossi	05/10/1979	genova	via scribanti	16	16131	34012345678
6635	paola	bianchi	12/04/1976	genova	via dodecaneso	35	16146	0104647992
6635	paola	bianchi	12/04/1976	genova	via dodecaneso	35	16146	0103536699
6635	paola	bianchi	12/04/1976	genova	via dodecaneso	35	16146	34812345678




determinare nome, cognome e numeri di telefono dei clienti (o di un cliente) è più efficiente

```
SELECT nome, cognome, telefono  
FROM CLIENTE
```

# possibile denormalizzazione

CLIENTE(codCli,*nome*,*cognome*,*dataN*,*città*,*via*,*no*,*cap*,telefono)

codCli	nome	cognome	dataN	città	via	no	cap	telefono
6610	anna	rossi	05/10/1979	genova	via scribanti	16	16131	01055664433
6610	anna	rossi	05/10/1979	genova	via scribanti	16	16131	34012345678
6635	paola	bianchi	12/04/1976	genova	via dodecaneso	35	16146	0104647992
6635	paola	bianchi	12/04/1976	genova	via dodecaneso	35	16146	0103536699
6635	paola	bianchi	12/04/1976	genova	via dodecaneso	35	16146	34812345678

 Ma, ad es., determinare l'età media dei clienti è meno efficiente  
(date di nascita più “sparpagate”)

La scelta dello schema dipende  
dalle operazioni da eseguire (workload)

# oltre la prima forma normale

Una relazione **nested relational** è una relazione che non è in prima forma normale (1NF)

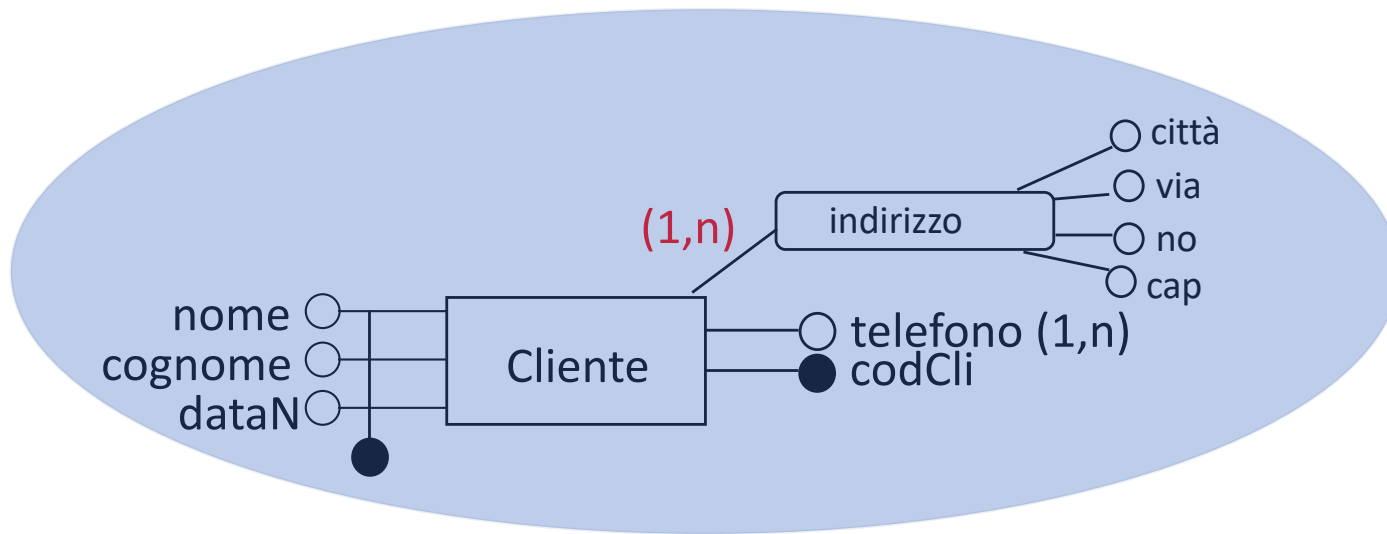
cioè può contenere gruppi di attributi (potenzialmente annidati) multi-valore

CLIENTE(codCli, nome, cognome, dataN, città, via, no, cap, (telefono) )

codCli	nome	cognome	dataN	città	via	no	cap	(telefono)
6610	anna	rossi	05/10/1979	genova	via scribanti	16	16131	(01055664433, 34012345678)
6635	paola	bianchi	12/04/1976	genova	via dodecaneso	35	16146	(0104647992, 0103536699, 34812345678)

Ogni cliente ha un certo numero di numeri di telefono (almeno uno)  
[per opzionalità manteniamo il pedice <sub>o</sub>]

e se volessi gestire anche più indirizzi?



Ristrutturazione standard

CLIENTE(codCli, nome, cognome, dataN)

RISPONDEA(codCli<sup>CLIENTE</sup>, telefono)

RECAPITO(codCli<sup>CLIENTE</sup>, città, via, no, cap)

# possibile denormalizzazione

CLIENTE(codCli,*nome*,*cognome*,*dataN*,città,via,no,cap,telefono)

codCli	nome	cognome	dataN	città	via	no	cap	telefono
6610	anna	rossi	05/10/1979	genova	via scribanti	16	16131	01055664433
6610	anna	rossi	05/10/1979	genova	via scribanti	16	16131	34012345678
6610	anna	rossi	05/10/1979	milano	via assietta	31	20161	01055664433
6610	anna	rossi	05/10/1979	milano	via assietta	31	20161	34012345678
6635	paola	bianchi	12/04/1976	genova	via dodecaneso	35	16146	0104647992
6635	paola	bianchi	12/04/1976	genova	via dodecaneso	35	16146	0103536699
6635	paola	bianchi	12/04/1976	genova	via dodecaneso	35	16146	34812345678
....								



ridondanza:

- nome, cognome, dataN sono ripetuti per ogni numero di telefono e ogni indirizzo
- ... in generale tutte le combinazioni ...



# abbandonando la prima forma normale

Una relazione **nested relational** è una relazione che non è in prima forma normale (1NF), cioè può contenere **gruppi di attributi** (potenzialmente annidati) multi-valore

CLIENTE(codCli, nome, cognome, dataN, (città, via, no, cap), (telefono) )

codCli	nome	cognome	dataN					(telefono)
				(città	via	no	cap)	
6610	anna	rossi	05/10/1979	(genova milano	via scribanti via assietta	16 31	16131, 20161)	(01055664433, 34012345678)
6635	paola	bianchi	12/04/1976	(genova	via dodecaneso	35	16146)	(0104647992, 0103536699, 34812345678)

# abbandonando la prima forma normale

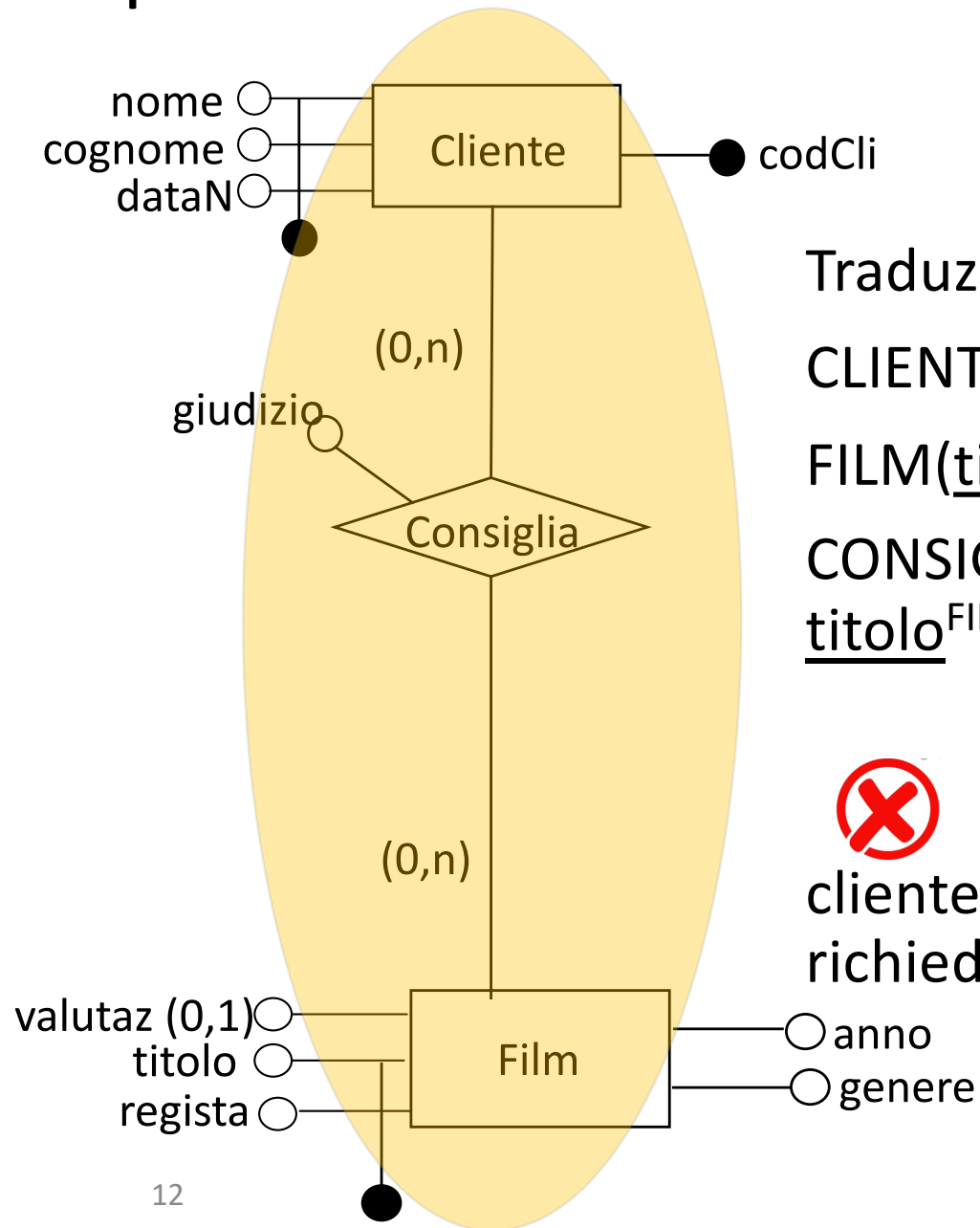
- Possiamo specificare un nome per gruppi di attributi multivalore
- CLIENTE(codCli, nome, cognome, dataN, indirizzi:(città, via, no, cap), (telefono) )

codCli	nome	cognome	dataN	indirizzi				(telefono)
				(città	via	no	cap)	
6610	anna	rossi	05/10/1979	(genova milano	via scribanti via assietta	16 31	16131, 20161)	(01055664433, 34012345678)
6635	paola	bianchi	12/04/1976	(genova	via dodecaneso	35	16146)	(0104647992, 0103536699, 34812345678)

in SQL? (ma noi non li usiamo!)

```
CREATE TABLE cliente  
(codCli DECIMAL(4),  
nome VARCHAR(20),  
dataN DATE,  
telefono SET (CHAR(15)),  
indirizzi SET (ROW (citta VARCHAR(20), via VARCHAR(20),  
no VARCHAR(10), cap INTEGER)))
```

# e per le associazioni?



Traduzione standard

CLIENTE(codCli, nome, cognome, dataN, ...)

FILM(titolo, regista, anno, genere, valutaz<sub>o</sub>)

CONSIGLIA(codCli<sup>CLIENTE</sup>,  
titolo<sup>FILM</sup>, regista<sup>FILM</sup>, giudizio)



Determinare nome, cognome del cliente e titolo e regista dei film consigliati richiede un join

# possibile denormalizzazione

CLIENTE(codCli,nome,cognome,dataN, titolo<sub>0</sub><sup>FILM</sup>,regista<sub>0</sub><sup>FILM</sup>,giudizio<sub>0</sub>)

codCli	nome	cognome	dataN	titolo	regista	giudizio
6610	anna	rossi	05/10/1979	underground	emir kusturica	3
6610	anna	rossi	05/10/1979	mediterraneo	gabriele salvatores	4
6635	paola	bianchi	12/04/1976	underground	emir kusturica	4
6635	paola	bianchi	12/04/1976	mediterraneo	gabriele salvatores	4
6635	paola	bianchi	12/04/1976	pulp fiction	quentin tarantino	4



ridondanza:

nome, cognome, ... sono ripetuti per film consigliato

# possibile denormalizzazione

CLIENTE(codCli,nome,cognome,dataN, titolo<sub>o</sub><sup>FILM</sup>,regista<sub>o</sub><sup>FILM</sup>,giudizio<sub>o</sub>)

codCli	nome	cognome	dataN	titolo	regista	giudizio
6610	anna	rossi	05/10/1979	underground	emir kusturica	3
6610	anna	rossi	05/10/1979	mediterraneo	gabriele salvatores	4
6635	paola	bianchi	12/04/1976	underground	emir kusturica	4
6635	paola	bianchi	12/04/1976	mediterraneo	gabriele salvatores	4
6635	paola	bianchi	12/04/1976	pulp fiction	quentin tarantino	4



determinare nome, cognome e i film consigliati è più efficiente

```
SELECT nome, cognome, titolo, regista  
FROM CLIENTE
```

# possibile denormalizzazione

CLIENTE(codCli,nome,cognome,dataN, titolo<sub>o</sub><sup>FILM</sup>,regista<sub>o</sub><sup>FILM</sup>,giudizio<sub>o</sub>)

codCli	nome	cognome	dataN	titolo	regista	giudizio
6610	anna	rossi	05/10/1979	underground	emir kusturica	3
6610	anna	rossi	05/10/1979	mediterraneo	gabriele salvatores	4
6635	paola	bianchi	12/04/1976	underground	emir kusturica	4
6635	paola	bianchi	12/04/1976	mediterraneo	gabriele salvatores	4
6635	paola	bianchi	12/04/1976	pulp fiction	quentin tarantino	4



ma, nuovamente, determinare l'età media dei clienti è meno efficiente

# abbandonando la prima forma normale

CLIENTE(codCli, nome, cognome, dataN, (titolo<sup>FILM</sup>, regista<sup>FILM</sup>, giudizio)<sub>0</sub>)

codCli	nome	cognome	dataN			
				(titolo	regista	giudizio)
6610	anna	rossi	05/10/1979	(underground mediterraneo	emir kusturica gabriele salvatores	3, 4)
6635	paola	bianchi	12/04/1976	(underground mediterraneo pulp fiction	emir kusturica gabriele salvatores quentin tarantino	4, 4, 4)



# abbandonando la prima forma normale

CLIENTE(codCli, nome, cognome, dataN,

valutazioni:(titolo<sup>FILM</sup>, regista<sup>FILM</sup>, giudizio)<sub>o</sub>)

codCli	nome	cognome	dataN	valutazioni		
				(titolo	regista	giudizio)
6610	anna	rossi	05/10/1979	(underground mediterraneo	emir kusturica gabriele salvatores	3, 4)
6635	paola	bianchi	12/04/1976	(underground mediterraneo pulp fiction	emir kusturica gabriele salvatores quentin tarantino	4, 4, 4)

# la scelta dipende dalle query (dal workload)

- Se la nostra esigenza fosse
- Determinare per ogni film il suo genere, l'anno e i giudizi ricevuti
- Con lo schema

FILM(titolo, regista, anno, genere, valutaz<sub>O</sub>, (codCli<sup>CLIENTE</sup>, giudizio)<sub>O</sub>)

l'interrogazione non richiede join e può essere eseguita efficientemente

# ulteriore nesting

- Se la nostra esigenza fosse
- Determinare per ogni film il suo genere, l'anno e i giudizi ricevuti, con nome e cognome del cliente che li ha formulati
- Con lo schema

FILM(titolo, regista, anno, genere, valutaz<sub>o</sub>, (nome,cognome, giudizio)<sub>o</sub>)

l'interrogazione non richiede join e può essere eseguita efficientemente

# Sulla base del workload, anche ulteriore ridondanza

- Q1: Età media dei clienti
- Q2: Nome e cognome dei clienti e relativi film consigliati
- Q3: Genere e anno dei film e relativi giudizi ricevuti, con nome e cognome del cliente che li ha formulati

CLIENTE(codCli, *nome*, *cognome*, *dataN*)

CLI\_GIUD(codCli, *nome*, *cognome*, *dataN*, (*titolo*<sup>FILM</sup>, *regista*<sup>FILM</sup>, *giudizio*)<sub>o</sub>)

FILM(titolo, regista, anno, genere, valutaz<sub>o</sub>, (*nome*, *cognome*, *giudizio*)<sub>o</sub>)

# Cosa vuol dire progettare lo schema logico in base a un workload?

- Per ogni interrogazione abbiamo uno schema logico che ne rende più efficiente l'esecuzione
- Ma lo schema logico è uno ...
- Dato un workload, guardiamo l'insieme di tutte le interrogazioni e individuiamo uno schema che permetta di eseguire in modo ragionevolmente efficiente le interrogazioni  
(rappresenta il miglior compromesso)
- Eventualmente introducendo ridondanza