Archivi e Basi di Dati

Memoria *persistente* e suo utilizzo

Bisogna organizzare i dati in modo da garantire

- inserimento
- cancellazione
- modifica e
- ricerca

su grandi quantita' di dati

DBMS

DataBase Management Systems:

sistemi informatici (programmi applicativi) per la gestione di collezioni di file che contengono informazioni coerenti fra loro

I DBMS devono essere trasparenti rispetto ai supporti fisici utilizzati

Modello (logico) piu' utilizzato: relazionale

Dimensioni ...

... anche enormi (terabytes).

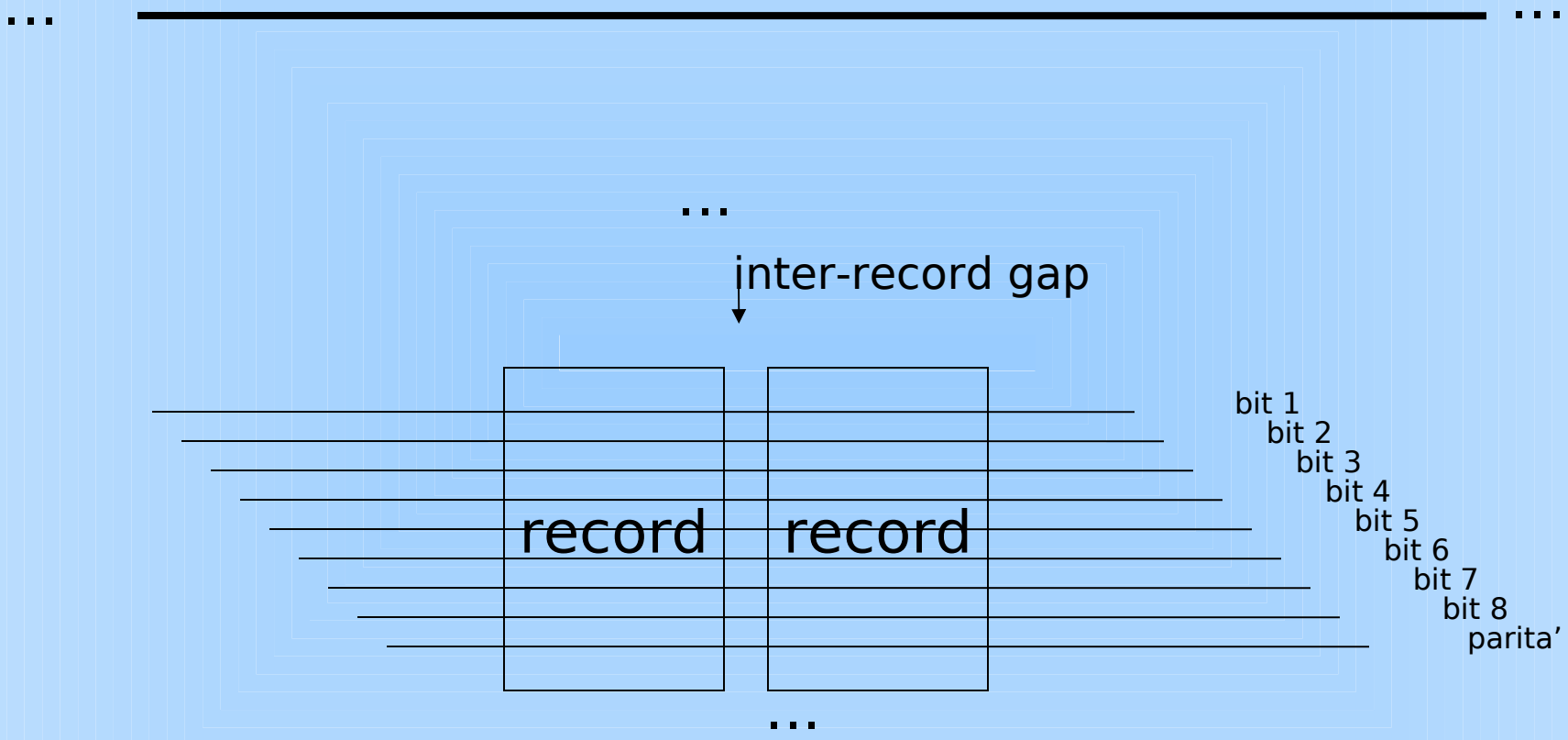
Supporti utilizzati: (soprattutto) nastri e dischi

Tecnologia magnetica

Nastri, accesso sequenziale, lenti, usati soprattutto per il backup

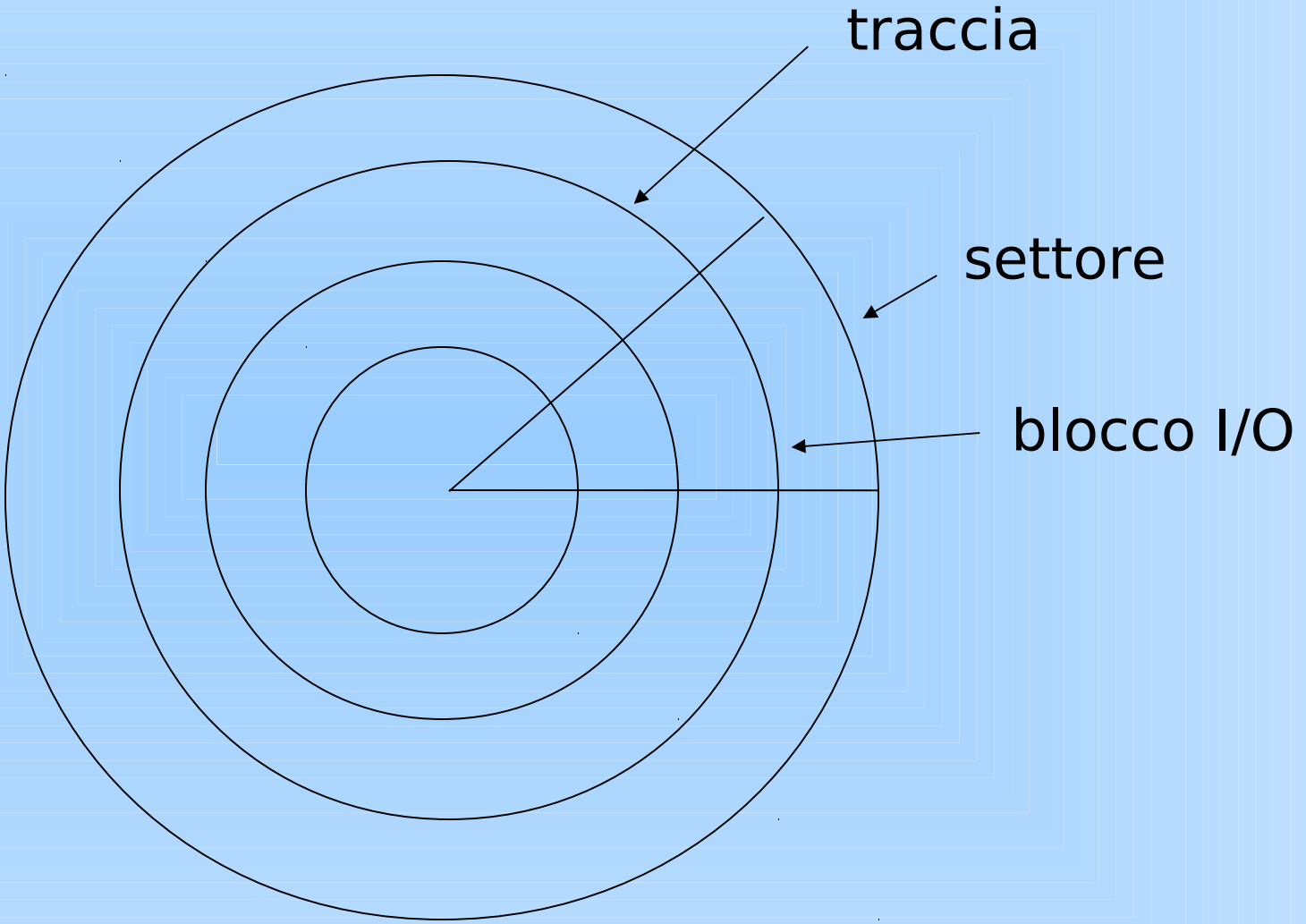
Come funzionano i nastri

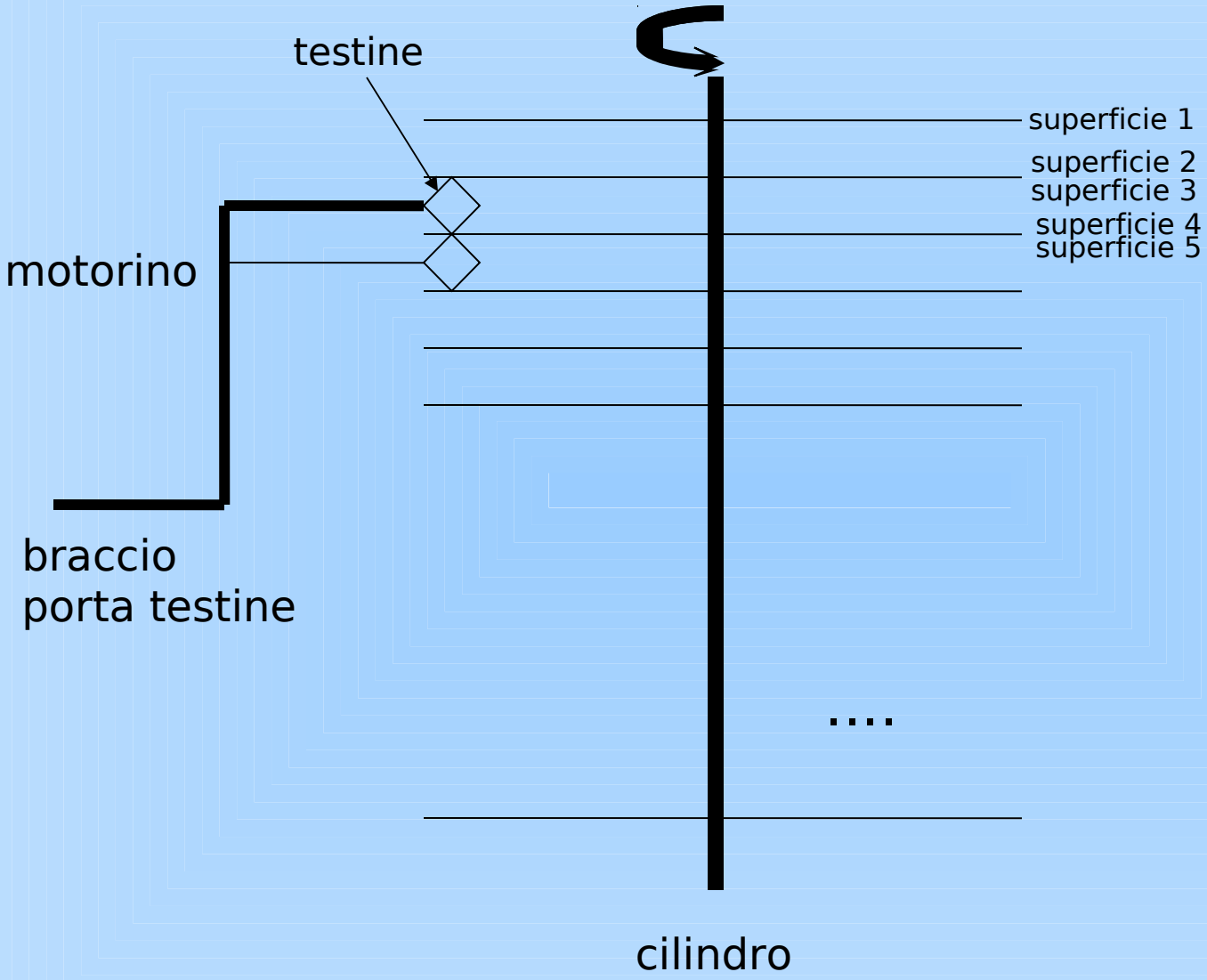
- piu' tracce (piu' tracce parallele lette a "spirale")
- densita: bpi (bit per inch)
- testina di lettura/scrittura che scorre sequenzialmente sul nastro
- bit di parita'
- record (registrazione): unita' componente di un file


...
...
...
inter-record gap
record
record
bit 1
bit 2
bit 3
bit 4
bit 5
bit 6
bit 7
bit 8
parita’
...
...
...

Dischi

- organizzazione di disck-pack: set di piatti (due superfici per piatto)
- tracce: cerchi concentrici
- settori
- suddivisione ottenuta mediante la formattazione (operazione “pericolosa”)


traccia
settore
blocco I/O


testine
superficie 1
superficie 2
superficie 3
superficie 4
superficie 5
motorino
braccio
porta testine
....
cilindro

... altri termini usati

- buffer
- tempo di seek
- tempo di latenza
- floppy disk

L’organizzazione *logica* dei dati

- record logici (i blocchi possono essere piu’ grandi o piu’ piccoli di un record logico)
- un file e’ costituito da una sequenza di record
- un record e’ costituito da un insieme di campi
- una base di dati e’ costituita da una collezione di file

Esempio

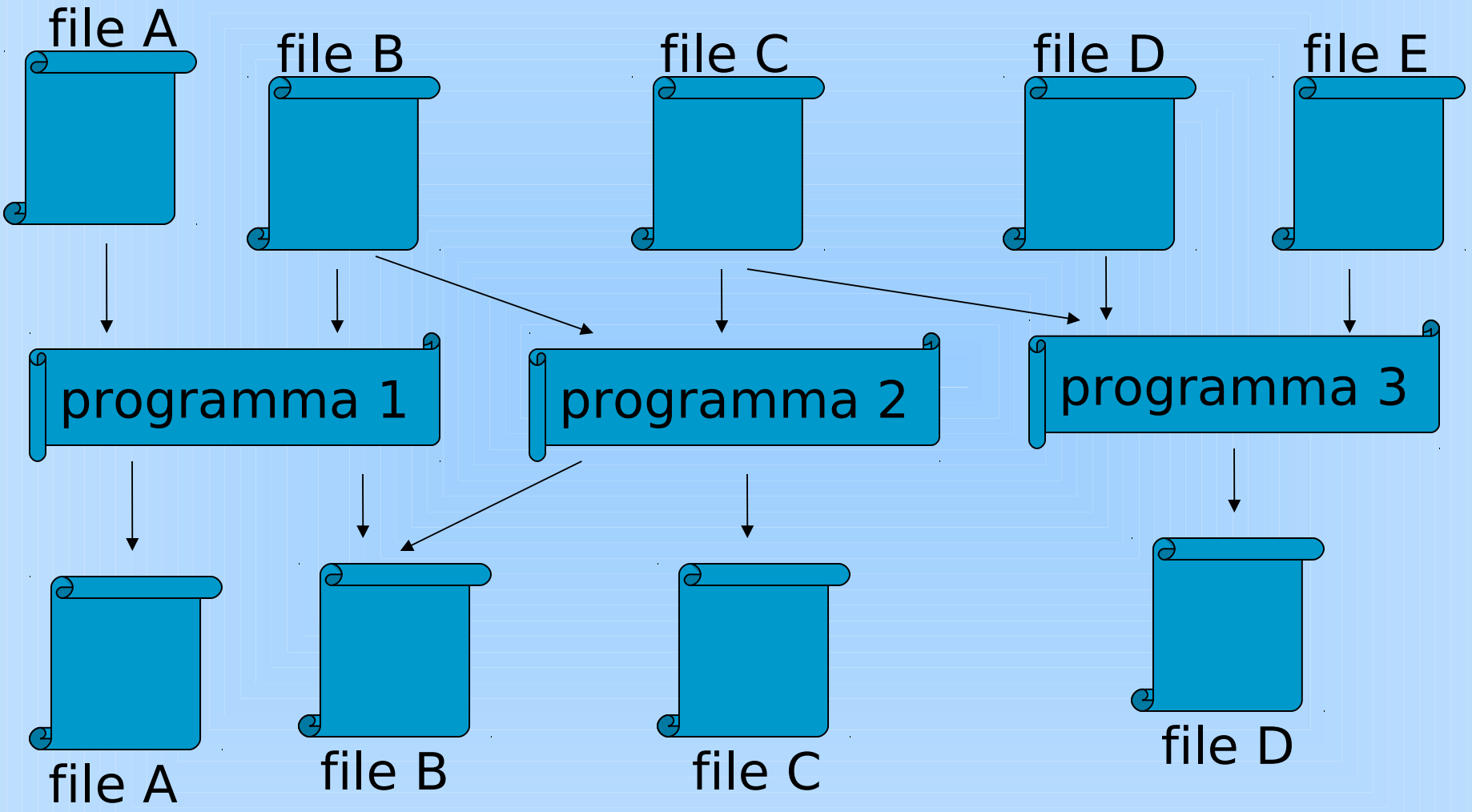
record	nome	indirizzo	telefono
1	A. Policriti	via delle Scienze 206	0432 558464
.....	.....	.....	.....

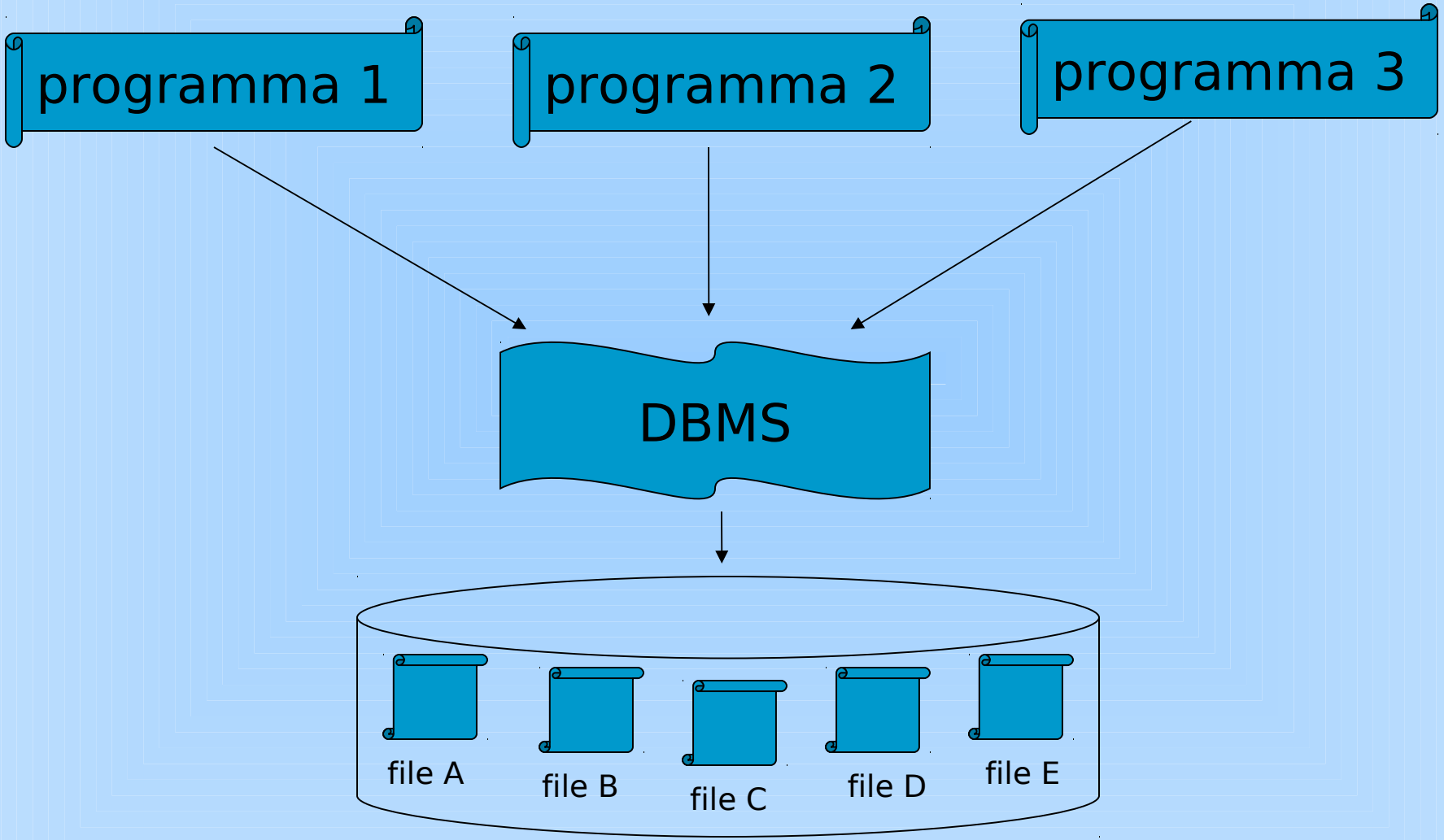
DBMS

- servono a *consultare* comodamente i dati in modo indipendente dal supporto fisico
- sembrano sono “semplici”: spesso usate dai non-specialisti
- ... a prima vista sembrerebbe che se ne puo’ fare a meno. Non e’ cosi’.

Cosa ci garantisce l'uso di un database

- Evitano inconsistenza e ridondanza dei dati
- Garantiscono la riservatezza dei dati
- Integrita' dei dati
- Accesso concorrente ai dati


file A
file B
file C
file D
file E
programma 1
programma 2
programma 3
file A
file B
file C
file D


programma 1
programma 2
programma 3
DBMS
file A
file B
file C
file D
file E

I DBMS permettono:

- di costruire diverse viste sui dati
- disciplinare l'accesso ad ogni campo di ogni record (vincoli di integrita')
- garantire accesso concorrente

Modelli (logici)

- modello gerarchico: il piu' vecchio
- modello reticolare: molto usato
- **modello relazionale: il piu' famoso (Codd 1970)**
- modello a oggetti: il piu' moderno e forse promettente

Terminologia a e linguaggi

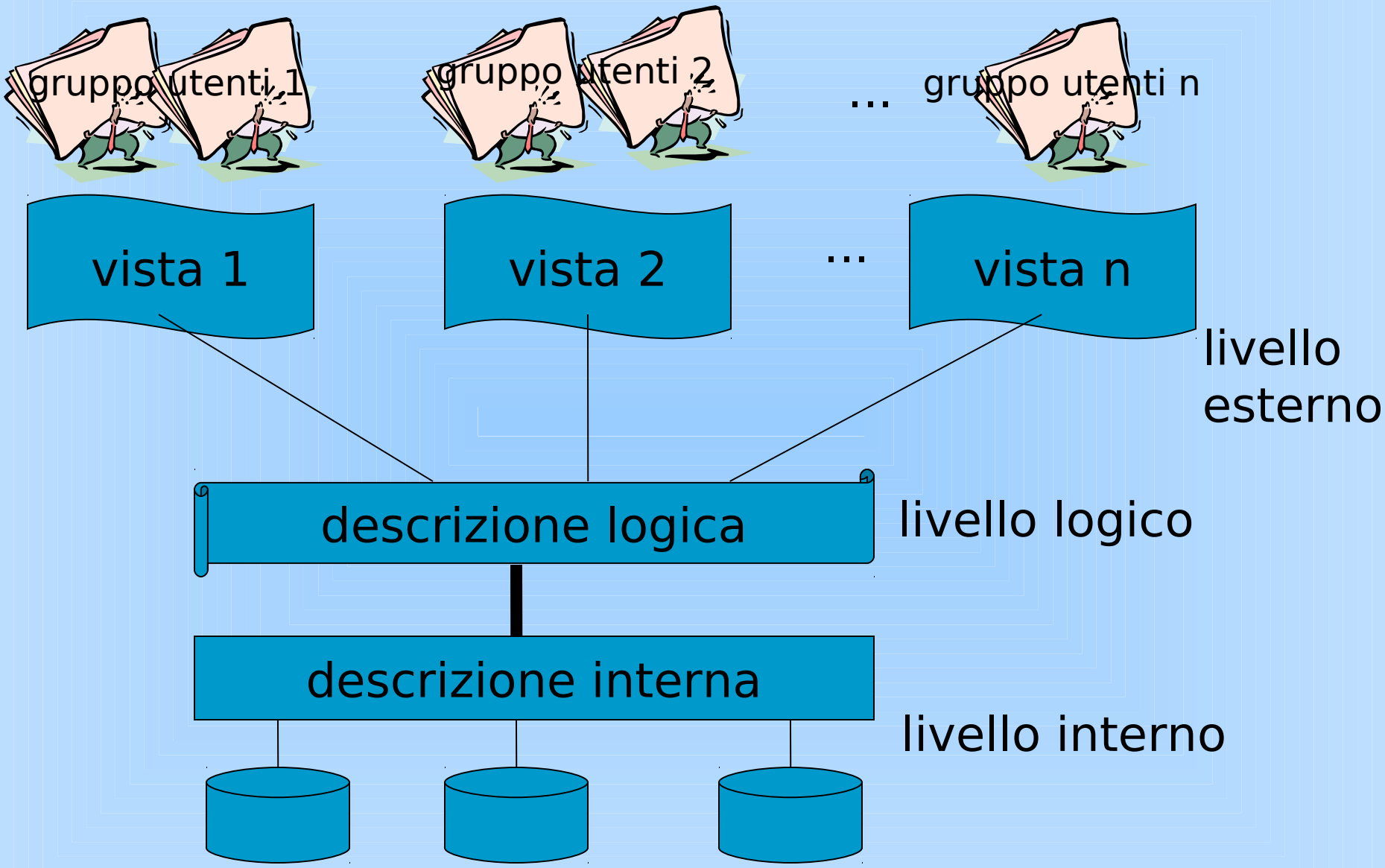
- Schema di una base di dati (disegno)
- Istanza o occorrenza di uno schema
- DDL (Data Definition Language): usato per definire i tipi e le operazioni su cui la base di dati operera'
- DML (Data Manipulation Language): usato per formulare le query (interrogazioni) e per modificare il contenuto (inserimenti cancellazioni modifica dei valori)

Livelli

Dal piu' basso al piu' alto sono:

- Fisico
- Logico
- Esterno

Bisogna garantire sia l'indipendenza fisica che logica nel disegno (tipi di dato astratto)


gruppo utenti 1
gruppo utenti 2
...
gruppo utenti n
vista 1
vista 2
...
vista n
livello esterno
descrizione logica
livello logico
descrizione interna
livello interno

ACID properties

1. Atomicita (ogni transazione esegue commit, ed eventalmente torna indietro rollback)
2. Consistenza (i vincoli di integrita' devono essere garantiti: rollback, rollback parziale)
3. Isolamento (ogni transazione "crede" di essere l'unica)
4. Durabilita' (persistenza delle transazioni)

Come si programma un DBMS

- DBA (Data Base Administrator): disegna e modifica la base di dati, usa il DDL
- Terminalisti: usa il linguaggio che gli mette a disposizione la vista
- Programmatori applicativi: usa il DML
- Utenti casuali: anche loro usano il loro DML

Basi di dati relazionali

- semplici
- generali
- ok anche su pc
- basate sul *modello* (logico) relazionale
- utilizzano spesso il linguaggio SQL come DDL e DML

Terminologia usata nelle basi di dati relazionali: corrispondenza logico-concettuale

Una base di dati e' una collezione di relazioni

Una relazione e' una collezione di entita'

- relazione e' sinonimo di tabella
- colonne della tabella: attributi (numero fisso, parte della definizione della relazione)
- righe della tabella: tuple (numero variabile, sono gli elementi dell'istanza della relazione)

- dominio di una colonna: tipo dei dati contenuti
- grado di una relazine: numero di colonne
- cardinalita’ di una relazione: numero di righe

Ricordiamo sempre la differenza tra schema e istanza di una relazione.

Esempio (classico)

Relation CONTO-CORRENTE
(NUMERO-CC: integer,
NOME: char(20),
INDIRIZZO: char(20),
SALDO: integer).

Relation MOVIMENTO
(NUMERO-CC: integer,
DATA-MOV: date,
NUMERO-MOV: integer,
IMPORTO: integer,
CAUSALE: char(1)).

CONTO-CORRENTE

NUMERO-CC	NOME	INDIRIZZO	SALDO
1	Rossi	v. Anemoni 5	3.678
2	Bianchi	v. Bolla 64	664
3	Brunelli	v. Po 41	6.777
4	Grandi	v. Romolo 3	3.400

MOVIMENTO

NUMERO-CC	DATA-MOV	NUMERO-MOV	IMPORTO	CAUSALE
1	14/1/02	1	+200	V
1	14/1/02	2	-500	P
1	27/1/02	1	+2.700	S
4	27/1/02	1	+1.850	S
3	25/1/02	1	-650	A

Operazioni delicate

- disegno dello schema *concettuale* (entita' relazione): bisogna prevedere tutto. *Data model*
- ricostruzione delle informazioni dalle informazioni in tabella (vedremo degli esempi: SQL al lavoro)
- il modello sembra semplice ma non lo e'

Archivi e Basi di Dati: Il linguaggio SQL

Esempio (classico)

Relation **CONTO-CORRENTE**

(NUMERO-CC: integer,
NOME: char(20),
INDIRIZZO: char(20),
SALDO: integer).

Relation **MOVIMENTO**

(NUMERO-CC: integer,
DATA-MOV: date,
NUMERO-MOV: integer,
IMPORTO: integer,
CAUSALE: char(1)).

CONTO-CORRENTE

NUMERO-CC	NOME	INDIRIZZO	SALDO
1	Rossi	v. Anemoni 5	3.678
2	Bianchi	v. Bolla 64	664
3	Brunelli	v. Po 41	6.777
4	Grandi	v. Romolo 3	3.400

MOVIMENTO

NUMERO-CC	DATA-MOV	NUMERO-MOV	IMPORTO	CAUSALE
1	14/1/02	1	+200	V
1	14/1/02	2	-500	P
1	27/1/02	1	+2.700	S
4	27/1/02	1	+1.850	S
3	25/1/02	1	-650	A

Nozioni importanti

- restrizione di una tupla t sugli attributi A (relativamente alla relazione R): t[A]
- chiave: sottoinsieme dell'insieme degli attributi che identifica univocamente la tupla e garantisce *unicita'* e *minimalita'*

Ex. trovare la chiave delle relazioni CONTO-CORRENTE e MOVIMENTO

NB la chiave non e' detto sia unica (la principale e' detta chiave primaria)

notazione per le chiavi

CONTO-CORRENTE (NUMERO-CC, NOME, INDIRIZZO, SALDO)

MOVIMENTO
(NUMERO-CC, DATA-MOV, NUMERO-MOV, IMPORTO, CAUSALE)

Il linguaggio SQL (Structured Query Language)

- Linguaggio presente in tutti i piu' importanti DBMS relazionali
- Definizione prodotta dall'ANSI e ISO
- SQL3 (1999/2000)
- sintassi complessa
- linguaggio ridondante

Istruzioni principali

- **CREATE TABLE** permette di creare una relazione e ne definisce attributi, vincoli e altro
- **PRIMARY KEY** definisce la chiave primaria

NB stiamo lavorando al livello *logico*

Indici

- identificano univocamente le tuple e permettono l’accesso mediante strutture dati appositamente disegnate
- la chiave primaria definisce implicitamente un indice *univoco*

Esempio (classico)

```
CREATE TABLE CONTO-CORRENTE
        (NUMERO-CC: integer, PRIMARY KEY, NOT NULL,
        NOME: char(20), NOT NULL,
        INDIRIZZO: char(20),
        SALDO: integer, NOT NULL).

CREATE UNIQUE INDEX CONTO-CORRENTE-KEY
ON CONTO-CORRENTE(NUMERO-CC)

CREATE TABLE MOVIMENTO
        (NUMERO-CC: integer, NOT NULL,
        DATA-MOV: date, NOT NULL,
        NUMERO-MOV: integer, NOT NULL,
        IMPORTO: integer,
        CAUSALE: char(1)
        PRIMARY KEY(NUMERO-CC, DATA-MOV, NUMERO-MOV)).

CREATE UNIQUE INDEX MOVIMENTO-KEY
ON MOVIMENTO(NUMERO-CC, DATA-MOV, NUMERO-MOV)
```

DDL instructions

CREATE TABLE

CREATE INDEX

DROP TABLE

DROP INDEX

.....

DML: queries

Le query vengono compilate usando i seguenti operatori (che manipolano tabelle):

- selezione
- proiezione
- join
- unione
- differenza

... nel linguaggio SQL ci sono istruzioni che "mimano" queste operazioni.

Le istruzioni in SQL sono rappresentate in *blocchi*

Le *clausole*

- Ogni blocco contiene uno dei seguenti tre tipi di clausole
  - SELECT opera sugli attributi
  - FROM opera sulle tabelle
  - WHERE esprime condizioni per la ricerca

inoltre

  - DISTINCT elimina i duplicati

Esempi

ragioniamo sulle tabelle dell'esempio della volta scorsa e introduciamo blocchi di istruzioni SQL che realizzino le operazioni fondamentali

- selezione (direttamente implementabile)
- proiezione
- join
- unione
- differenza

Una sola relazione

SELECT NOME, INDIRIZZO

FROM CONTO-CORRENTE

Operazione di *selezione* che produce come risultato una *proiezione* della tabella CONTO-CORRENTE sugli attributi NOME e INDIRIZZO

CONTO-CORRENTE

NUMERO-CC	NOME	INDIRIZZO	SALDO
1	Rossi	v. Anemoni 5	3.678
2	Bianchi	v. Bolla 64	664
3	Brunelli	v. Po 41	6.777
4	Grandi	v. Romolo 3	3.400

SELECT NOME, INDIRIZZO

FROM CONTO-CORRENTE

NOME	INDIRIZZO
Rossi	v. Anemoni 5
Bianchi	v. Bolla 64
Brunelli	v. Po 41
Grandi	v. Romolo 3

CONTO-CORRENTE

NUMERO-CC	NOME	INDIRIZZO	SALDO
1	Rossi	v. Anemoni 5	3.678
2	Bianchi	v. Bolla 64	664
3	Brunelli	v. Po 41	6.777
4	Grandi	v. Romolo 3	3.400

SELECT SALDO

FROM CONTO-CORRENTE

WHERE NUMERO-CC = 2

SALDO
664

CONTO-CORRENTE

NUMERO-CC	NOME	INDIRIZZO	SALDO
1	Rossi	v. Anemoni 5	3.678
2	Bianchi	v. Bolla 64	664
3	Brunelli	v. Po 41	6.777
4	Grandi	v. Romolo 3	3.400

MOVIMENTO

NUMERO-CC	DATA-MOV	NUMERO-MOV	IMPORTO	CAUSALE
1	14/1/02	1	+200	V
1	14/1/02	2	-500	P
1	27/1/02	1	+2.700	S
4	27/1/02	1	+1.850	S
3	25/1/02	1	-650	A

SELECT NOME, INDIRIZZO

FROM CONTO-CORRENTE, MOVIMENTO

WHERE DATA-MOV = 27/1/02 **AND** CONTO-CORRENTE.NUMERO-CC = MOVIMENTO.NUMERO-CC

NOME	INDIRIZZO
Rossi	v. Anemoni 5
Grandi	v. Romolo 3

SELECT NOME, INDIRIZZO

FROM CONTO-CORRENTE **JOIN** MOVIMENTO

ON CONTO-CORRENTE.NUMERO-CC = MOVIMENTO.NUMERO-CC

WHERE DATA-MOV = 27/1/02

alternativa (ridondanza)

Queries binarie

CONTO-CORRENTE

NUMERO-CC	NOME	INDIRIZZO	SALDO
1	Rossi	v. Anemoni 5	3.678
2	Bianchi	v. Bolla 64	664
3	Brunelli	v. Po 41	6.777
4	Grandi	v. Romolo 3	3.400

MOVIMENTO

NUMERO-CC	DATA-MOV	NUMERO-MOV	IMPORTO	CAUSALE
1	14/1/02	1	+200	V
1	14/1/02	2	-500	P
1	27/1/02	1	+2.700	S
4	27/1/02	1	+1.850	S
3	25/1/02	1	-650	A

SELECT NUMERO-CC
FROM CONTO-CORRENTE
WHERE SALDO > 2.000
UNION
SELECT NUMERO-CC
FROM MOVIMENTO
WHERE IMPORTO > 1.000

NUMERO-CC
1
3
4

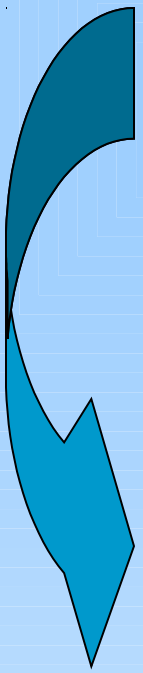
NB gli schemi delle tabelle devono essere compatibili

CONTO-CORRENTE

NUMERO-CC	NOME	INDIRIZZO	SALDO
1	Rossi	v. Anemoni 5	3.678
2	Bianchi	v. Bolla 64	664
3	Brunelli	v. Po 41	6.777
4	Grandi	v. Romolo 3	3.400

MOVIMENTO

NUMERO-CC	DATA-MOV	NUMERO-MOV	IMPORTO	CAUSALE
1	14/1/02	1	+200	V
1	14/1/02	2	-500	P
1	27/1/02	1	+2.700	S
4	27/1/02	1	+1.850	S
3	25/1/02	1	-650	A



SELECT NUMERO-CC

FROM CONTO-CORRENTE

WHERE SALDO > 2.000

MINUS

SELECT NUMERO-CC

FROM MOVIMENTO

WHERE IMPORTO > 1.000

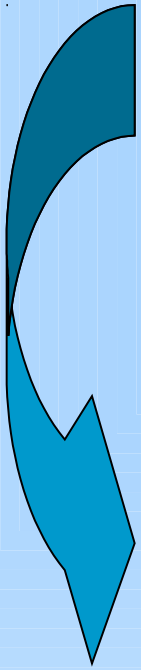
NUMERO-CC
3

CONTO-CORRENTE

NUMERO-CC	NOME	INDIRIZZO	SALDO
1	Rossi	v. Anemoni 5	3.678
2	Bianchi	v. Bolla 64	664
3	Brunelli	v. Po 41	6.777
4	Grandi	v. Romolo 3	3.400

MOVIMENTO

NUMERO-CC	DATA-MOV	NUMERO-MOV	IMPORTO	CAUSALE
1	14/1/02	1	+200	V
1	14/1/02	2	-500	P
1	27/1/02	1	+2.700	S
4	27/1/02	1	+1.850	S
3	25/1/02	1	-650	A



SELECT DATA-MOV, NUMERO-MOV, IMPORTO, CAUSALE

FROM MOVIMENTO, CONTO-CORRENTE

WHERE NOME = Rossi

AND CONTO-CORRENTE.NUMERO-CC = MOVIMENTO.NUMERO-CC

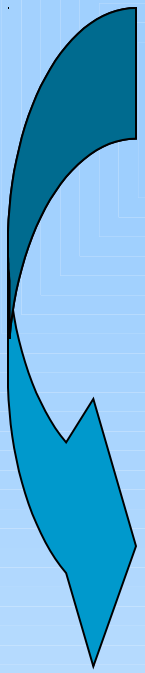
DATA-MOV	NUMERO-MOV	IMPORTO	CAUSALE
14/1/02	1	+200	V
14/1/02	2	-500	P
27/1/02	1	+2.700	S

CONTO-CORRENTE

NUMERO-CC	NOME	INDIRIZZO	SALDO
1	Rossi	v. Anemoni 5	3.678
2	Bianchi	v. Bolla 64	664
3	Brunelli	v. Po 41	6.777
4	Grandi	v. Romolo 3	3.400

MOVIMENTO

NUMERO-CC	DATA-MOV	NUMERO-MOV	IMPORTO	CAUSALE
1	14/1/02	1	+200	V
1	14/1/02	2	-500	P
1	27/1/02	1	+2.700	S
4	27/1/02	1	+1.850	S
3	25/1/02	1	-650	A



SELECT NOME, IMPORTO, CAUSALE

FROM MOVIMENTO, CONTO-CORRENTE

WHERE SALDO > 2.000

AND (CAUSALE = V **OR** CAUSALE =S)

AND CONTO-CORRENTE.NUMERO-CC = MOVIMENTO.NUMERO-CC

NOME	IMPORTO	CAUSALE
Rossi	+200	V
Rossi	+2.700	S
Grandi	+1.850	S

CONTO-CORRENTE

NUMERO-CC	NOME	INDIRIZZO	SALDO
1	Rossi	v. Anemoni 5	3.678
2	Bianchi	v. Bolla 64	664
3	Brunelli	v. Po 41	6.777
4	Grandi	v. Romolo 3	3.400

MOVIMENTO

NUMERO-CC	DATA-MOV	NUMERO-MOV	IMPORTO	CAUSALE
1	14/1/02	1	+200	V
1	14/1/02	2	-500	P
1	27/1/02	1	+2.700	S
4	27/1/02	1	+1.850	S
3	25/1/02	1	-650	A

... voglio il
10% dell'importo

SELECT NUMERO-CC, DATA-MOV, NUMERO-MOV, IMPORTO * O.1

FROM MOVIMENTO

WHERE CAUSALE =S

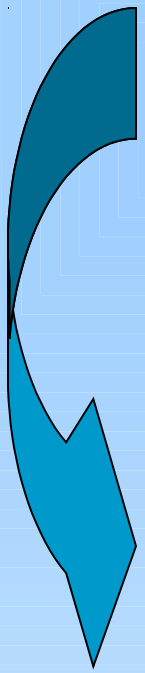
NUMERO-CC	DATA-MOV	NUMERO-MOV	IMPORTO*0.1
1	27/1/02	1	+270
4	27/1/02	1	+185

CONTO-CORRENTE

NUMERO-CC	NOME	INDIRIZZO	SALDO
1	Rossi	v. Anemoni 5	3.678
2	Bianchi	v. Bolla 64	664
3	Brunelli	v. Po 41	6.777
4	Grandi	v. Romolo 3	3.400

MOVIMENTO

NUMERO-CC	DATA-MOV	NUMERO-MOV	IMPORTO	CAUSALE
1	14/1/02	1	+200	V
1	14/1/02	2	-500	P
1	27/1/02	1	+2.700	S
4	27/1/02	1	+1.850	S
3	25/1/02	1	-650	A



SELECT NOME, SALDO, IMPORTO, CAUSALE

FROM MOVIMENTO, CONTO-CORRENTE

WHERE SALDO + 50 < -IMPORTO

AND CAUSALE = A

AND CONTO-CORRENTE.NUMERO-CC = MOVIMENTO.NUMERO-CC

risultato vuoto!

Estrarre l'intera relazione CONTO-CORRENTE ordinata in modo crescente rispetto a SALDO

```
SELECT *
FROM CONTO-CORRENTE
ORDER BY SALDO
```

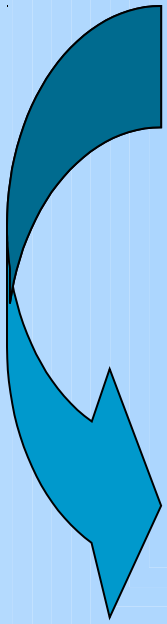
... decrescente

```
SELECT *
FROM CONTO-CORRENTE
ORDER BY SALDO DESC
```

combinare ordinamenti

MOVIMENTO

NUMERO-CC	DATA-MOV	NUMERO-MOV	IMPORTO	CAUSALE
1	14/1/02	1	+200	V
1	14/1/02	2	-500	P
1	27/1/02	1	+2.700	S
4	27/1/02	1	+1.850	S
3	25/1/02	1	-650	A



SELECT *

FROM MOVIMENTO

ORDER BY NUMERO-CC **ASC**, DATA-MOV **DESC**, NUMERO-MOV **ASC**

NUMERO-CC	DATA-MOV	NUMERO-MOV	IMPORTO	CAUSALE
1	27/1/02	1	+2.700	S
1	14/1/02	1	-500	V
1	14/1/02	2	+200	P
3	25/1/02	1	-650	A
4	27/1/02	1	+1.850	S

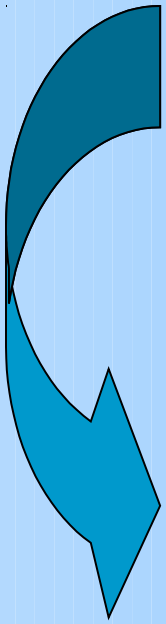
Grouping

L’operazione di raggruppamento genera classi (di equivalenza) rispetto al valore di uno o piu’ attributi

Una volta generati i gruppi e’ possibile introdurre condizioni sui gruppi oppure operare con
SUM, AVG, MIN, MAX

MOVIMENTO

NUMERO-CC	DATA-MOV	NUMERO-MOV	IMPORTO	CAUSALE
1	14/1/02	1	+200	V
1	14/1/02	2	-500	P
1	27/1/02	1	+2.700	S
4	27/1/02	1	+1.850	S
3	25/1/02	1	-650	A



SELECT NUMERO-CC, SUM(IMPORTO), COUNT(*)

FROM MOVIMENTO

WHERE DATA-MOV > 1/1/02 **AND** DATA-MOV < 31/1/02

GROUP BY NUMERO-CC

HAVING SUM(IMPORTO) > 1.000

NUMERO-CC	SUM(IMPORTO)	COUNT(*)
1	+2.400	3
4	+1.850	1

Blocchi innestati

```
SELECT NOME, INDIRIZZO
FROM CONTO-CORRENTE
WHERE NUMERO-CC IN
        SELECT  NUMERO-CC
        FROM MOVIMENTO
        WHERE DATA-MOV = 27/1/02
```

si valutano i blocchi dal piu’ interno al piu’ esterno

Inserimento, cancellazione e modifica

INSERT INTO CONTO-CORRENTE **VALUES**
(<5, Cannetti, c. Tirolo 5, 100>)

DELETE FROM MOVIMENTO
WHERE CAUSALE = S

UPDATE CONTO-CORRENTE
SET SALDO = SALDO * 1.001
WHERE NOME = Bianchi

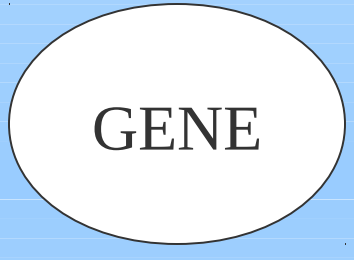
ACID properties

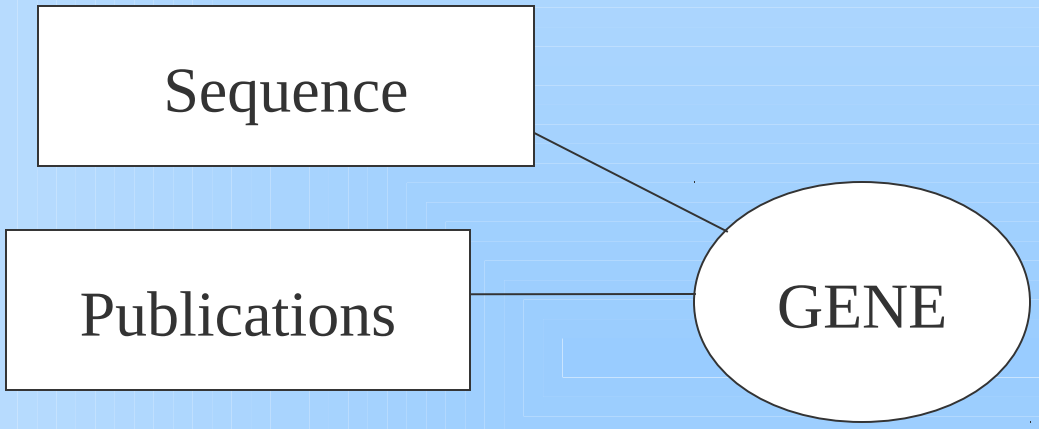
Ogni transazione deve terminare con un COMMIT WORK o un ROLLBACK WORK
(se non lo fa l'utente o l'applicazione, lo fa il sistema

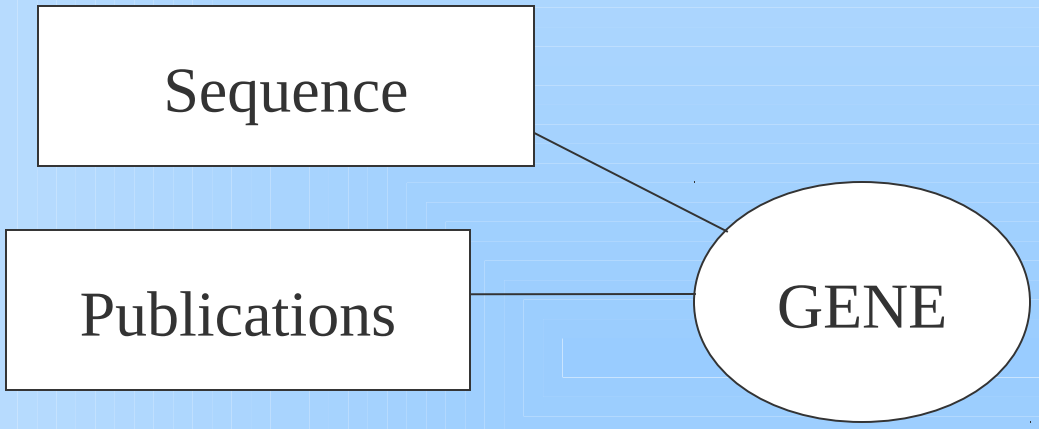
```
UPDATE CONTO-CORRENTE
SET SALDO = SALDO + 500
WHERE NOME = Bianchi;
UPDATE CONTO-CORRENTE
SET SALDO = SALDO - 500
WHERE NOME = Rossi;
```

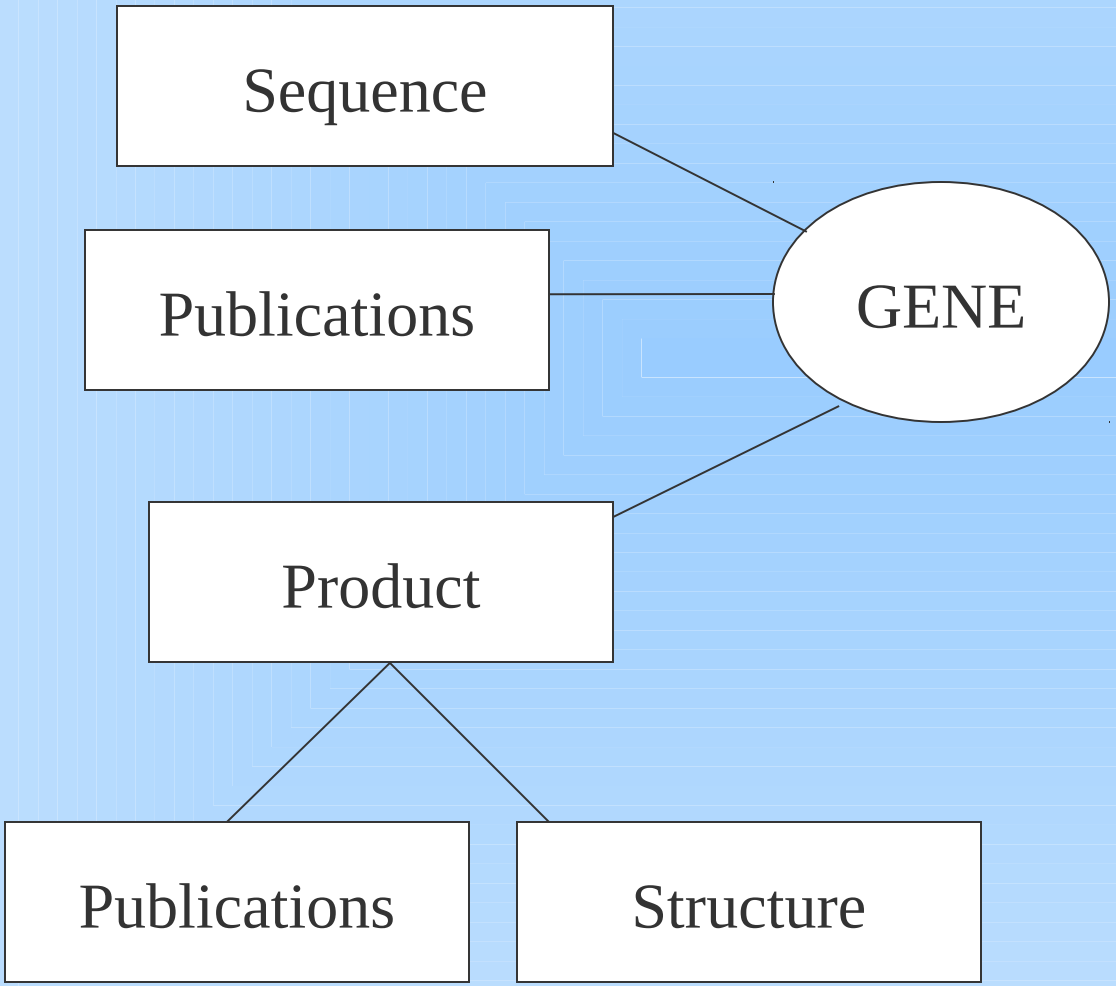
il trasferimento avviene

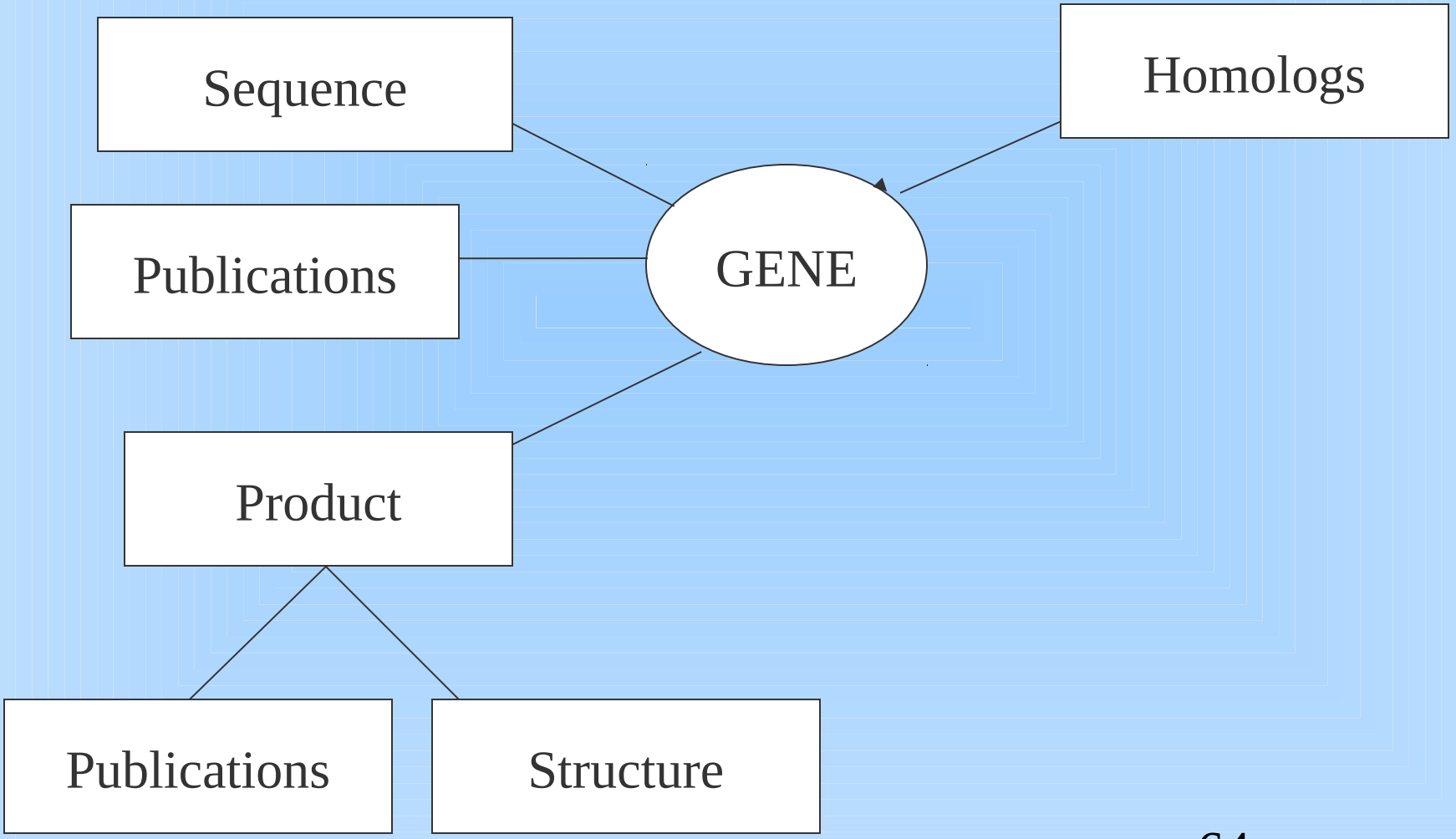
tutto come prima

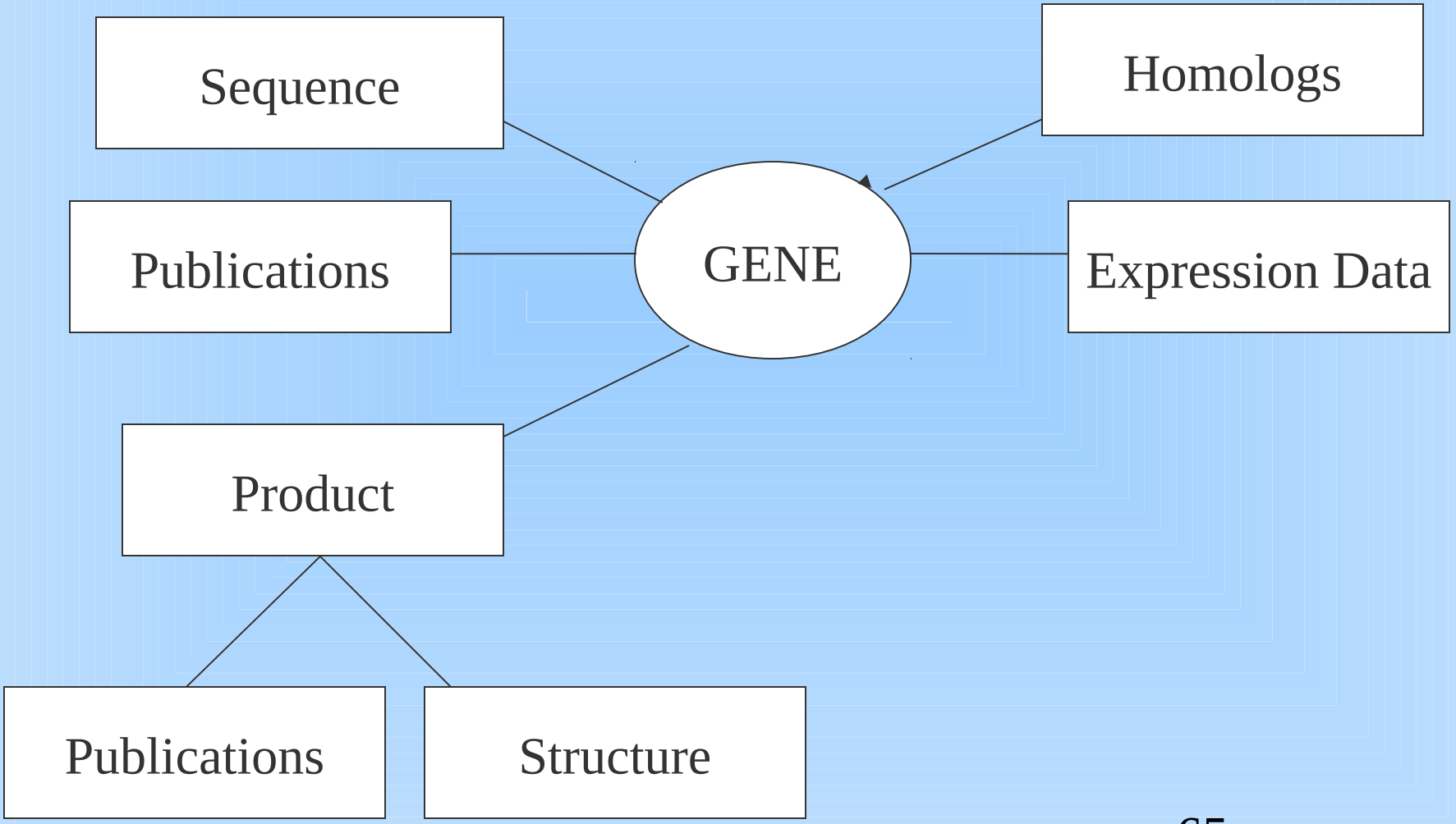

GENE

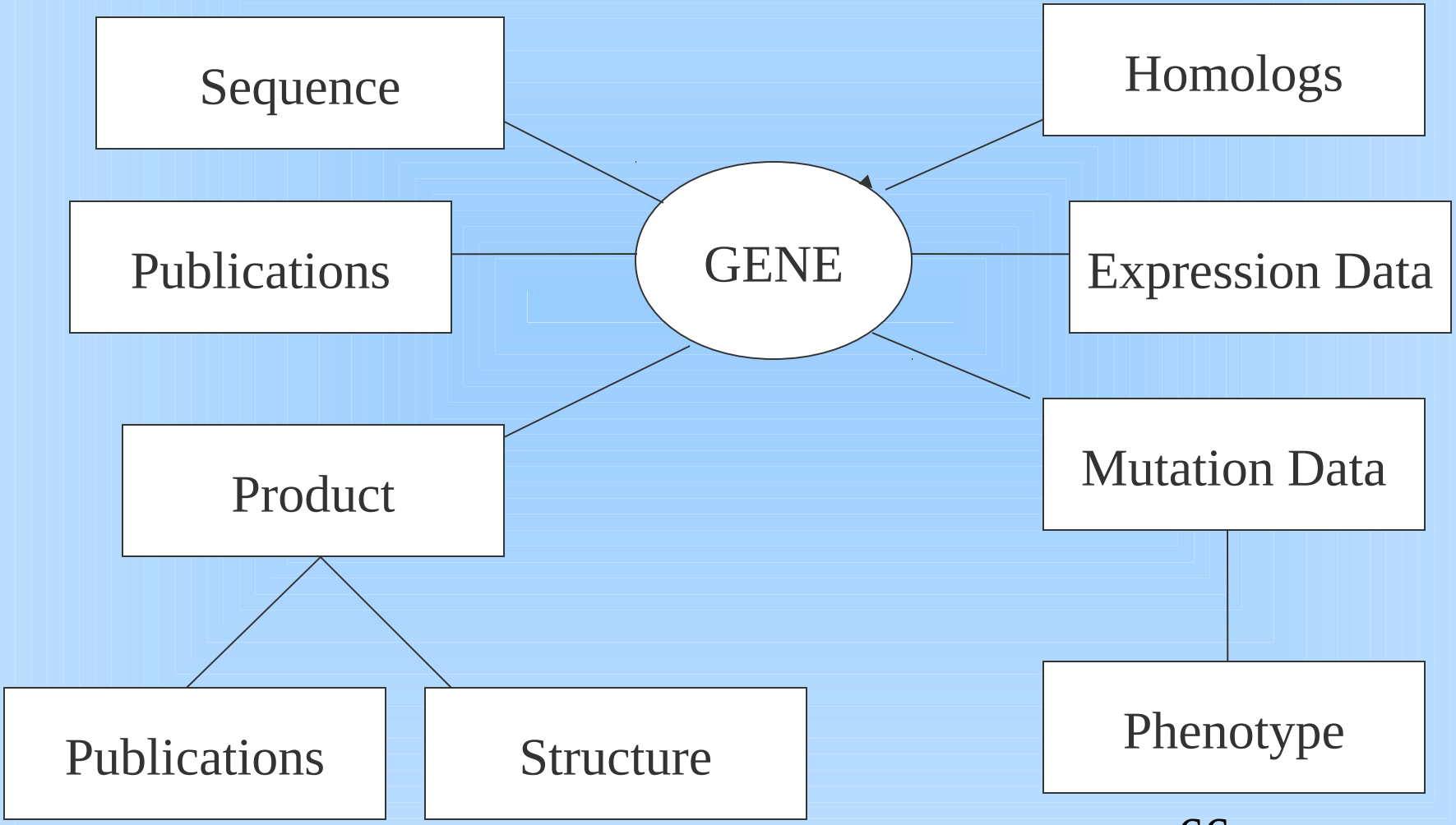

Sequence
Publications
GENE


Sequence
Publications
GENE


Sequence
Publications
GENE
Product
Publications
Structure


Sequence
Homologs
Publications
GENE
Product
Publications
Structure


Sequence
Homologs
Publications
GENE
Expression Data
Product
Publications
Structure


Sequence
Homologs
Publications
GENE
Expression Data
Product
Mutation Data
Publications
Structure
Phenotype