UNIVERSITA' DEGLI STUDI DI PADOVA

FACOLTA' DI SCIENZE STATISTICHE

CORSO DI LAUREA IN STATISTICA E GESTIONE DELLE IMPRESE

TESI DI LAUREA

SVILUPPO DI SOFTWARE PER LA MISURA DEL RISCHIO DI MERCATO PER PORTAFOGLI NON LINEARI NEI FATTORI DI RISCHIO

Relatore: ch.mo PROF M. BONOLLO

Laureando: PAOLO PESSOLI

PROGETTAZIONE DI BASI DI DATI RELAZIONALI E NORMALIZZAZIONE

Introduzione alle basi di dati

Di solito, le situazioni del mondo reale non sono organizzate in modo bidimensionale.
Basti pensare, infatti, al numero dei clienti che ordinano un solo articolo, o a quello degli impiegati che si occupano di un unico progetto, oppure a quanti sono gli studenti che seguono un solo corso.
In realtà, si vive in un mondo tridimensionale fatto di relazioni "uno-a-molti" perciò, mentre due dimensioni potrebbero anche essere sufficienti in un foglio elettronico, nei database si hanno strutture più rigide e un'ulteriore dimensione che costituisce un grande vantaggio.
Paragonando una tabella ad un foglio elettronico, ovverosia facendo corrispondere i record alle righe ed i campi alle colonne, anche un database organizzato in modo non relazionale offre un modello bidimensionale della realtà, consentendone quindi una visione non stereoscopica. Un database relazionale, invece, aggiunge la terza dimensione perciò, se si rapportasse il tutto ai concetti di lunghezza, altezza e profondità, la riga corrisponderebbe all'altezza, la colonna alla lunghezza e la relazione alla profondità.
La maggior parte del lavoro svolto, direttamente o indirettamente, in un database si concentra sulla tabella che può essere dunque considerata come il suo cuore. Il fatto che nel database vi siano diverse tabelle correlate tra loro non implica però che queste siano pronte a garantirne le funzionalità tipiche, giacché nella progettazione di un database occorre considerare quanto segue:

- Lo scopo del database.
- Le tabelle ed i campi occorrenti.
- Le relazioni necessarie tra le tabelle.
- Come evitare la ridondanza dei dati.
- Come garantire la consistenza dei dati.

Esaminando queste voci si comprendono rapidamente i vantaggi apportati dalla normalizzazione dei dati, ovverosia dal *processo mediante il quale si semplifica il progetto di un database per ottenerne la massima efficienza ed uniformità*.
Per alcuni la normalizzazione è *il processo che consente di individuare ed eliminare da un database i dati ridondanti e le relative anomalie, mentre per altri è il processo con cui si suddividono le tabelle in parti più piccole, in modo da ottimizzarle*.
Di fatto, una giusta combinazione di entrambe le definizioni è quanto di più vicino alla realtà giacché, per garantire una buona progettazione del database ed un'adeguata struttura di tabella, occorre evitare a tutti i costi entrambi i vizi di progettazione, ovverosia la **rindondanza** e l'**inconsistenza**.

Si usa l'espressione *ridondanza interna* per fare riferimento alla ridondanza presente all'interno di una tabella, mentre l'espressione *ridondanza esterna* per fare riferimento alla duplicazione della stessa informazione, esclusi i campi chiave, in più di una tabella. La ridondanza esterna è comune nei fogli elettronici, poiché questi non impediscono l'immagazzinamento della stessa informazione in più tabelle, fogli o cartelle di lavoro.
La ridondanza crea anomalie e queste creano a loro volta inconsistenze.
Si supponga, per esempio, che alcuni numeri d'ordine non siano associati ad alcun cliente, come avverrebbe se la tabella Clienti non fosse collegata e si modificasse il codice di un cliente, o se si eliminassero le sue informazioni dalla tabella Ordini; in questo caso, gli ordini del cliente risulterebbero "dati persi" vaganti e senza alcun collegamento che servirebbero solo ad occupare spazio. È per questa ragione che l'ordine dipende dal cliente.
Un altro tipo di inconsistenza si crea quando si immettono gli stessi dati in più posizioni, giacché si potrebbe ad esempio scrivere correttamente il nome di una entità in una circostanza, ma non nell'altra. In questo senso, si può dire che la normalizzazione riduce al minimo le occasioni in cui si può verificare un errore umano.
Vi sono dunque buone probabilità che occorra normalizzare una tabella contenente un numero eccessivo di campi o una gran quantità di dati ridondanti poiché, suddividendola in tabelle più piccole e definendo i collegamenti tra quelle correlate, si otterrebbero i seguenti vantaggi:

- Eliminazione dei dati duplicati, sia all'interno della stessa tabella che tra quelle correlate.
- Conseguimento della massima velocità ed efficienza dal database.
- Garanzia che non si debba immettere la stessa informazione più di una volta.
- Risparmio dello spazio su disco mediante l'eliminazione dei dati ridondanti.
- Certezza della consistenza e dell'uniformità dei dati immessi e prodotti.
- Creazione di un progetto di più facile manutenzione e manipolazione rispetto ad uno con dati non normalizzati.
- Realizzazione di un meccanismo per ricerche condizionali nelle **query**, nelle **maschere** e nei **report**.

Nel momento in cui si immagazzina più volte un dato nella stessa tabella o in più tabelle, si crea dunque una *ridondanza di dati*, ma perché occorre evitarla durante la progettazione di un database? Se l'indirizzo di un fornitore fosse ripetuto in 50 righe di dati e questi si trasferisse da Milano a Roma, bisognerebbe modificare tutte le righe, dando luogo a ciò che si definisce *un'anomalia di aggiornamento.*
Per peggiorare le cose, se in un modello non relazionale si cancellasse una parte correlata a quel fornitore, si eliminerebbero anche i suoi dati, dando luogo a ciò che si definisce *un'anomalia in cancellazione.*

Le anomalie in cancellazione ed aggiornamento sono una diretta conseguenza della ridondanza dei dati, ma per fortuna esiste un approccio migliore che consente di ottenere un progetto ottimale del database.

CONCETTI FONDAMENTALI DELLA NORMALIZZAZIONE

Il modello relazionale dei dati è stato introdotto da E.F. Codd (un ricercatore dell'IBM) fin dal 1970 ed è basato sul concetto matematico *di relazione fra insiemi.* Il fatto di essere nato all'interno di un rigoroso ambito matematico ha consentito una formulazione del modello completamente libera da considerazioni di tipo pratico pertinenti all'efficienza dell'implementazione.

Proprio a causa dei problemi di efficienza si è dovuto attendere diversi anni prima di disporre dei primi prototipi dì DBMS relazionali: fra questi il più noto è quasi certamente il *System* R ultimato dall'IBM solo nel 1978 e rivelatosi comunque inefficiente per grosse basi di dati.

A partire dai primi anni '80 si è però assistito ad una crescente diffusione sul mercato di sistemi relazionali anche se limitati alla gestione di piccole e medie basi di dati (diciamo dell'ordine *delle* migliaia di registrazioni). Questo sviluppo è stato favorito anche dalla diffusione dei «personal» utilizzati appunto da organizzazioni con un volume di dati relativamente modesto.

Anche la teoria delle basi di dati relazionali si è notevolmente sviluppata al punto da sminuire l'importanza dei modelli gerarchico e reticolare. Ciò ha consentito di mettere a punto delle tecniche di implementazione che, congiuntamente all'impiego di hardware specializzato *(database machines,* dischi associativi, ecc.), rendono sempre più appetibile l'uso di DBMS relazionali anche per grosse basi di dati.

Definendo il modello del database relazionale E. F. Codd stabilì una serie progressiva di regole, dette *forme normali,* da applicare al database per ottenere gradualmente una struttura ottimale ed una migliore progettazione complessiva.

Le Forme Normali vanno usate primariamente come strumento ausiliario di verifica della qualità degli schemi di relazione prodotti e non come metodologia a se stante di progettazione.

Gli schemi di relazione vengono prodotti (mapping dallo schema concettuale E-R) a partire dalle Entità e dalle Relationship che compaiono nello schema concettuale, se la fase di progettazione concettuale è stata effettuata in modo corretto, gli schemi di relazione ottenuti dal mapping risultano naturalmente normalizzati

In questo contesto, la teoria della normalizzazione risulta comunque utile, come strumento di verifica degli schemi risultanti dall'attività sia di *progettazione logica* sia di *progettazione concettuale*.

Vincoli di chiave

- *SUPERCHIAVE :* una superchiave di uno schema di relazione è un insieme di attributi X che, in ogni istanza valida di uno schema di relazione, i valori degli attributi in X individuano univocamente un'ennupla, ovvero tale che in nessuna istanza valida dello schema di relazione possano esistere due ennuple diverse che coincidono su tutti gli attributi in X.
- *CHIAVE :* una chiave di uno schema di relazione è una superchiave minimale, nel senso che se si elimina un attributo, i rimanenti non formano più una superchiave.

Un attributo che appartiene ad una chiave è chiamato *attributo primo*.

- *CHIAVE PRIMARIA :* è una delle chiavi dello schema di relazione, e di solito viene preferita quella con il minor numero di attributi.
- *CHIAVE ESTERNA :* detti A e B due tipi di entità ad S un'associazione 1:N da A a B, ci proponiamo di trovare uno schema relazionale in grado di rappresentare A, B ed S. La soluzione classica prevede di introdurre due relazioni così con concepite :

  $\Rightarrow$ Una relazione R_A avente gli attributi di A

  $\Rightarrow$ Una relazione R_B avente gli attributi di B e gli attributi chiave di R_A

Gli attributi chiave di R_A presenti in R_B costituiscono una cosiddetta **chiave esterna**, cioè una chiave appartenente ad un'altra relazione (appunto "esterna").

Il valore di una chiave esterna rappresenta un puntatore simbolico (o logico) alla ennupla di una certa relazione ed è l'equivalente dei puntatori fisici (nascosti alla vista dell'utente): attraverso il valore di una chiave esterna, un'ennupla di una relazione viene associata a quell'ennupla della relazione riferita che ha il valore della chiave primaria uguale al valore della chiave esterna.

Dipendenze funzionali

La teoria delle basi di dati relazionali fa costante riferimento al concetto di *dipendenza funzionale* come uno strumento formale con cui è possibile:

- esprimere dei vincoli di integrità intra-relazionali
- giudicare della qualità, *forma normale*, di uno schema
- decomporre schemi in sottoschemi ottimali

Una dipendenza funzionale, denotata da $X \rightarrow Y$, tra due gruppi di attributi X e Y di uno schema R specifica un **vincolo** sulle le possibili tuple che possono formare una istanza legale r di R e si legge:

X determina funzionalmente Y

Y dipende funzionalmente da X

Sia $X = \{X_1, X_2, \ldots X_k\}$ un insieme di attributi di una relazione R e Y un attributo di R, dire che $X \rightarrow Y$ significa asserire che i valori della componente Y dipendono da, sono determinati da, i valori della componente X, o in altre termini, se per uno schema R sussiste $X \rightarrow Y$, allora ogni volta che due tuple in r (R) coincidono per i valori su X, esse debbono coincidere anche per i valori su Y.

Si osserva che, per definizione, se X è una chiave di uno schema R ogni altro attributo di R dipende funzionalmente da X.
Dire che X → Y significa asserire che i valori della componente Y dipendono da, sono determinati dai valori della componente X.
Da X → Y non necessariamente discende Y → X.
Ad esempio, asserire per un dato schema la df :

Prof → Corso

significa asserire che, nella particolare realtà modellata dallo schema, sussiste il vincolo "*un Professore può tenere un solo Corso*"
Ad esempio, la df :

Stud, Materia → Tutor

significa che uno Studente ha un unico Tutor per ogni determinata Materia, mentre

Stud → Tutor

significa che ad uno Studente viene assegnato un unico Tutor indipendentemente dalla Materia

Una *dipendenza funzionale* è una caratteristica dello schema, *intensione*, e non della particolare istanza dello schema, *estensione* ed è dettata dalla semantica degli attributi di uno schema e non può essere inferita da una *istanza* di relazione.
Una istanza di uno schema che rispetti una data dipendenza funzionale viene detta *estensione legale* dello schema rispetto alla data dipendenza funzionale.
E' compito del progettista della base di dati specificare per un dato schema R l'insieme dei vincoli F, l'insieme, cioè, delle dipendenze funzionali dettate dalla semantica che gli attributi hanno nella particolare realtà modellata dallo schema.

CONDIZIONE **SUFFICIENTE** PER LA SCOMPOSIZIONE SENZA PERDITA DI INFORMAZIONE: una relazione R si decompone senza perdita di informazione su due relazioni R1 e R2 se l'insieme degli **attributi comuni** ad R 1 e R2 è **chiave** per almeno una delle due relazioni decomposte

Non tutte le scomposizioni effettuate per normalizzare le relazioni sono "buone", nel senso che soddisfano alle due proprietà essenziali di una scomposizione

- scomposizione senza perdita
- conservazione delle dipendenze

SCOMPOSIZIONE SENZA PERDITA: (Lossless Join property) significa che deve essere sempre possibile ricostruire la relazione originaria mediante *join naturale* delle relazioni ottenute dalla scomposizione
CONSERVAZIONE DELLE DIPENDENZE : significa che tutte le dipendenze definite sulla relazione originaria devono poter essere verificabili direttamente anche sulle relazioni ottenute nella scomposizione.

In generale l'insieme delle DF definite su di uno schema di relazione ha una struttura complessa: la separazione delle DF (e quindi dei concetti che esse rappresentano) non é facilitata dalla natura delle dipendenze stesse che risultano essere non naturalmente separabili e **dipendenti** l'una dall'altra; in tal caso: può non essere possibile basare la scomposizione su tutte le dipendenze e può essere difficile individuare quelle su cui si deve basare la scomposizione.
La teoria relazionale sulle dipendenze funzionali consente di individuare insiemi di DF (copertura minimale) equivalenti all'insieme originario tali che in essi le DF sono naturalmente separabili e tra loro indipendenti, perciò facilmente utilizzabili nella scomposizione di relazioni non rispondenti alla forma normale.

Prima forma normale

La prima ***forma normale*** prevede che **non vi siano gruppi di ripetizione** (colonne duplicate), vale a dire che non vi siano più campi appartenenti alla stessa categoria, dal momento che ognuno di essi corrisponde per lo più ad una categoria.
Secondariamente è prevista l'individuazione della **chiave primaria** semplice o composta, ossia un campo, o un insieme di campi, che identifica univocamente ogni record presente nella tabella: la prima forma normale consente di stabilire una chiave primaria, ma il fatto che si tratti di una chiave composta indica che non si è ancora ottenuta la seconda forma normale.
Infine è possibile attuare un processo di **suddivisione della tabella in altre più piccole** per ottenere una maggiore efficienza ed integrità dei dati.

Seconda forma normale

Una tabella deve trovarsi nella prima forma normale, prima di poter passare alla seconda forma normale. Un altro requisito è che tutte le colonne non incluse nella chiave, ovverosia le colonne secondarie, devono dipendere completamente dall'intera chiave primaria. In altri termini i campi secondari devono dipendere dalla chiave primaria composta in modo completo e **non parziale**.
Formalizzando: uno schema di relazione R è in Seconda Forma Normale se ogni attributo
non-primo dipende in modo *pieno*, *non parziale*, da ogni chiave dello schema R

Si ha una *dipendenza parziale o dipendenza funzionale parziale* quando questa è determinata non dall'*intera* chiave primaria ma solo da una sua *parte,* perciò è molto frequente imbattersi in dipendenze parziali e quindi in una prima forma normale, quando si ha una chiave composta.
Una relazione è *automaticamente* nella seconda forma normale, qualora contenga un solo attributo, non composto, nella propria chiave primaria.
Non rientra quindi nella analisi della dipendenza funzionale parziale il campo di collegamento per stabilire la relazione uno-a-molti tra due tabelle, detto in modo più specifico *chiave esterna* e appartenente quindi all'associazione.

Terza forma normale

Nella terza forma normale si ricercano le colonne secondarie che non dipendono funzionalmente dalla chiave primaria della tabella, ma da attributi non-primi ovverosia le *dipendenze transitorie*.
Una relazione R è in terza forma normale quando, per ogni possibile chiave di R:

- R è in seconda forma normale, cioè non esistono attributi non chiave che dipendono parzialmente dalla chiave
- Non esistono attributi non chiave che dipendono transitivamente dalla chiave

Formalmente: uno schema di relazione R è in terza forma normale se per ogni dipendenza funzionale $X \rightarrow A$ definita su R:

- o X è una *superchiave* dello schema R
- o A è un *attributo primo* dello schema R

Occorre prestare attenzione a non confondere le associazioni con le dipendenze, e quindi anche in questo caso non rientrano nella analisi della dipendenza funzionale le *chiavi esterne* costituenti le associazioni.
Come si può intuire, ogni schema relazionale in seconda forma normale può sempre essere messo in terza forma normale facendo "migrare" in relazioni separate le dipendenze fra attributi non chiave.

Forma normale di Boyce-Codd (BCNF)

Relazioni in Forma Normale (BCNF) evitano ridondanze e anomalie perchè tengono separati i ***concetti*** di interesse, uno per ogni relazione
E' possibile, in alcuni casi, scomporre una relazione che non soddisfa la forma normale in due o più relazioni normalizzate proiettando la relazione non normalizzata sugli **insiemi** di attributi che corrispondono a **singoli distinti concetti**
Come regola pratica:

- a partire dall'insieme delle Dip. Fun. definite sulla relazione non in BCNF, **formare gruppi distinti** di Dip. Fun. ciascuno dei quali è costituito da tutte le Dip. Fun. che hanno la parte determinante (sinistra) uguale

- proiettare la relazione non normalizzata su ciascuno insiemi di attributi corrispondente a ciascun gruppo di Dip. Fun. cosi formato

Nella relazione esempio risultano i tre gruppi di Dip. Fun.:

(1) IMPIEGATO → STIPENDIO

(2) PROGETTO→ BILANCIO

(3) {IMPIEGATO, PROGETTO} → RUOLO

La Dipendenza Funzionale :

{IMPIEGATO, PROGETTO} → RUOLO

è soddisfatta senza che vi siano due tuple eguali sulla sua parte determinante, ciò avviene perchè la **parte determinante** della Dipendenza Funzionale è una **superchiave** della relazione.

Generalizzando: ***ridondanze*** e ***anomalie*** sono causate dalle Dip. Fun. X → Y che consentono la presenza di più tuple fra loro uguali sugli attributi in X cioè tali che X non contiene una chiave.

Formalmente: una relazione è in Forma Normale di Boyce-Codd se per ogni Dip. Fun. X → Y definita sulla relazione, X è **superchiave** della relazione.

IL DATABASE REALIZZATO: E2E MARKET RISk

MODELLI DI ANALISI «VALUE AT RISK»

Engineering Ingegneria Informatica S.p.A.

Engineering Ingegneria Informatica è una società appartenente al Gruppo Engineering, leader sul mercato nazionale nella *system* e *business integration* e nei servizi di *application management* alle imprese, Engineering interviene sullo sviluppo di nuove logiche di business integrando le tecnologie tradizionali e le nuove frontiere di Internet con le singole strutture aziendali.

Core business del Gruppo è la realizzazione di progetti e di architetture innovative ad elevato contenuto tecnologico, ma sono i servizi di *application management* la componente più dinamica delle attività: tali servizi rappresentano più del 30% del fatturato globale.

Engineering nasce a Padova il 6 giugno 1980 con la denominazione di Cerved Engineering.

La società viene rilevata dai soci fondatori mediante un'operazione di *management buy out* nel 1984 e la Cerved esce. Nell'85 viene costituita Softlab, oggi la maggiore delle 10 controllate.

Agli inizi degli anni '90, la società avvia una fase di espansione: entrano nel capitale in qualità di investitori il Gruppo Paribas, Italmobiliare e IBM Italia che usciranno successivamente.

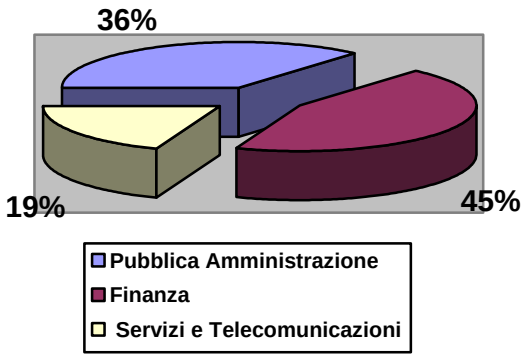
La fine degli anni '90 ed il primo scorcio degli anni 2000, coincidono con il consolidamento di Engineering sul mercato. La spinta alla crescita viene sostenuta da due scelte strategiche: la decisione di quotare la capogruppo al Nuovo Mercato e la costituzione di nuove società controllate.

Nasce Engiweb.com e viene acquisita Olivetti Sanità, poi Engisanità, oggi Engineering sanità Enti Locali.

Il 12 dicembre 2000 Engineering sbarca a Piazza Affari: entrano nel capitale gli attuali investitori stabili creditizi ed industriali che detengono il 7% del capitale sociale.

Parallelamente alla crescita dimensionale, il Gruppo si riposiziona sul mercato: i progetti di *system* e *business integration* ed i servizi di *application management* sono rivolti alle imprese di medie e grandi dimensioni, al mondo della Pubblica Amministrazione e al mercato emergente delle Utilities.

Il settore Finanza è storicamente il primo mercato su cui opera il Gruppo, seguito dalla Pubblica Amministrazione (Centrale e Locale, Difesa e Spazio, Sanità), dall'Industria e dalle Telecomunicazioni.



Attualmente il Gruppo, che nel 2003 ha un reso utile netto del 7,2 milioni di €[1], è costituito da:

- 9 società controllate;
- 3000 dipendenti di cui 1500 nella capogruppo Engineering Ingegneria Informatica;
- 30 sedi;
- 4 laboratori di Ricerca e Sviluppo (Roma, Napoli, Palermo e Dublino);
- 70 ricercatori.

L' offerta di Engineering, quindi, è rivolta a grandi clienti e organizzazioni complesse pubbliche e private impegnate nell'opera di ammodernamento delle strutture produttive.

Il prototipo E2E Market Risk: definizione degli obiettivi

Il prototipo E2E nasce dalla collaborazione di *Engineering* e *Euros Consulting* dal progetto di realizzare software per la misura e gestione del rischio di mercato.

Questo deriva dalla possibilità che le variabili di mercato (tassi di cambio, tassi di interesse, prezzi di azioni e merci, liquidità) possano muoversi in modo tale da rendere negativo il valore del

[1] Dati di Bilancio approvati al 30 settembre 2003.

contratto detenuto in portafoglio dall'istituzione finanziaria: l'obiettivo del software è di misurare i rischi di mercato generati dagli *Strumenti Finanziari* che vengono scambiati su mercati regolamentati ed OTC tramite il calcolo del *Value at Risk*, grandezza generalmente accettata come indice rappresentativo in questo ambito, secondo le metodologie sotto descritte.
In secondo luogo è previsto l'utilizzo dei fattori di rischio (prezzi di azioni e indici, tassi di interesse, tassi di cambio, prezzi delle materie prime) come mattoni principali per ogni funzione di pricing e in generale di valutazione del rischio: la funzione di pricig segue l'approccio Black & Scholes.
Per quanto riguarda i modelli di riferimento per la misurazione del rischio di mercato verranno utilizzati rispettivamente:
- Il modello parametrico per la misurazione dei rischi lineari
- Il modello Black & Scholes per la misurazione dei rischi non lineari.

Introduzione al Value At Risk

Le procedure per la misurazione di *value at risk* (VaR) di un portafoglio sono, attualmente, strumenti sempre più utilizzati, in un contesto dove da un lato vi sono degli operatori che necessitano d'informazioni facili da recuperare e da utilizzare, mentre dall'altro vi è una letteratura ricca ma complessa e soprattutto priva di un punto di riferimento comune. Esistono le più disparate metodologie per stimare la perdita potenziale attesa di un portafoglio con una certa probabilità prefissata, la scelta, da parte nostra, di analizzare alcune metodologie e di trascurarne altre è dipesa dal fatto che si è cercato, dapprima a livello teorico ed in un secondo momento attraverso la verifica empirica, di conciliare due esigenze tra loro conflittuali

- L'operatività e la maneggevolezza delle metodologie statistiche-matematiche da adottare per la stima del *value at risk* di un portafoglio;
- L'aderenza delle stesse alla realtà.

L'importanza della misurazione del VaR vanno viste non solo ex post per valutare il rischio cui è sottoposto, un portafoglio ma anche ex ante per l'acquisizione di un'insieme di strumenti finanziari che abbiano basso valore a rischio, oppure per la scelta della composizione di un portafoglio che abbia un .prefissato livello di rischiosità.
Di seguito considereremo il valore a rischio di un portafoglio azionario volgendo l'attenzione a metodologie basate sulla correlazione esistente tra diversi titoli, quale

a) **Matrice delle varianze/covarianze**

e metodologie basate sul **Capital Asset Pricing Model** (CAPM), quali

b) **RiskMetrics**

c) Adattamenti del modello «b» ai criteri suggeriti dal **Comitato di Basilea** di vigilanza bancaria;

Prima di procedere alla fase d'analisi dei singoli modelli si ritiene opportuno chiarire il concetto di distribuzione normale, ipotesi sulla quale si basa la costruzione delle formalizzazioni matematico-statistiche delle procedure di stima di un portafoglio azionario su elencate.
L'ipotesi fondamentale, sottostante il CAPM e il modello basato sul calcolo delle matrici delle varianze/covarianze, consta nel. considerare i rendimenti, semplici o logaritmici (dati rispettivamente dalle seguenti formule $(P_t - P_{t-1})/P_t$ e $\ln (P_t / P_{1-t})$), distribuiti normalmente, il che implica poter affermare che con una certa probabilità i rendimenti varieranno e i x deviazioni standard rispetto al loro valore medio. Spesso la distribuzione dei rendimenti ha un grado di curtosi diverso dalla gaussiana ed è asimmetrica, considerarla normale porterebbe a degli errori in fase di stima. L'eventualità di sottostimare il rischio può dipendere da una funzione di densità asimmetrica negativa e/o più schiacciata rispetto alla normale, (si sovrastimerà nei casi contrari). Con un grado di curtosi maggiore (minore) di tre, quindi rendimenti meno (più) concentrati attorno al valore medio (rispetto a una normale) e/o con asimmetria negativa (positiva), quindi maggiore (minore) probabilità in corrispondenza di rendimenti negativi si devono considerare deviazioni standard maggiori (minori) d 1.65 e 2.33 se si vuole effettuare la stima sotto l'ipotesi della probabilità unilaterale pari rispettivamente al 95% e al 99%.
Quando si parla di livello di probabilità s'intende sottinteso unilaterale, poiché a noi interessa solo la probabilità che non si verifichino perdite al di sotto del valore stimato, e non che vi siano utili maggiori di quelli attesi.
Tutti i modelli considerati si basano su ipotesi più o meno forti, in altre parole su assunzioni che potrebbero non essere completamente verificate nella realtà in periodi «turbolenti»; da ciò discende la possibilità di ottenere delle formalizzazioni matematiche dei modelli semplici e maneggevoli ma allo stesso tempo vi è l'eventualità che si commettano degli errori nella fase di stima. È possibile minimizzare gli errori di stima costruendo un portafoglio che abbia delle caratteristiche ben precise, oppure si possano ottenere delle valutazioni significative utilizzando delle tecniche di correzione del VaR.

L'analisi svolta e le scelte operative

Non sempre, se si vuole svolgere velocemente e con semplicità l'analisi di stima del *value at risk*, senza incorrere in elevati costi di ricerca dei dati, si può seguire quanto suggerito dalla teoria. Come in questo paragrafo è chiarito, anche noi in fase operativa abbiamo dovuto sottostare ai limiti imposti dal reperimento dei dati. Gli studi sono stati accentrati su

A) sistemi basati sulla teoria di Capitai Asset Pricing Model (CAPM), cioè modelli che utilizzano la relazione esistente tra l'indice di mercato e i titoli azionari, relazione di tipo lineare che vede l'utilizzo di un coefficiente β:

- RiskMetrics;
- Adattamenti del modello RiskMetrics ai criteri suggeriti dal Comitato di Basilea di vigilanza bancaria;

B) sistemi basati sul calcolo della matrice di correlazione tra tutti i titoli azionari che compongono il portafoglio

- Matrice delle varianze/covarianze.

Sono stati individuati, in maniera da ottenere un portafoglio diversificato, azioni appartenenti a diversi settori di mercato che presentavano differenti gradi di rischio di mercato e di rischio specifico.

Sono state prese in esame le serie storiche degli ultimi due anni (dal 23/07/2001 al 22/07/2003) dei prezzi delle azioni dei seguenti 6 titoli: Air Liquide, Lvmh, Engineering, Generali, Telecom, Fiat.

Per la stima del value at risk con il modello RiskMetrics e quelli ottenuti con varianti ai criteri dal primo suggerito è stato necessario individuare un indice di mercato di riferimento.

Per la semplicità di reperimento dei dati sono stati identificati 4 indici di riferimento: Mib30, Cac40, Euronext100, DJ Euro STOXX 50.

La necessità di individuare un indice di riferimento è dovuta alla teoria del Capital Asset Pricing Model che esprime il rischio di un portafoglio in funzione della volatilità di un indice di mercato e del coefficiente beta.

Beta, in parole molto semplici, non è altro che il coefficiente angolare della retta di regressione tra i rendimenti di mercato e quelli di una specifica azione, quindi è in grado di dirci quanto varia un titolo a fronte di una determinata variazione dell'indice di mercato. Beta, quasi sempre stimato in maniera lineare, può assumere teoricamente infiniti valori, ma solitamente i valori oscillano intorno all'unità.

Se $|\beta| > 1$ la variabilità dei titoli è più che proporzionale di quella di mercato, se

$|\beta| < 1$ la variabilità dei titoli è meno che proporzionale di quella di mercato, se

$|\beta| = 1$ la variabilità dei titoli è identica a quella di mercato, con

$\beta > 0$ il titolo ha attitudine a variare nella stessa direzione del mercato con

$\beta < 0$ l'attitudine delle azioni a variare è opposta alla direzione di mercato.

Di conseguenza è possibile distinguere le azioni in funzione del beta in aggressive ($\beta > 1$), neutre ($\beta = 1$) e difensive ($\beta < 1$).

La scelta dell'ampiezza dell'arco temporale è stata suggerita dalla letteratura e vincolata alla possibilità di reperire la serie storiche all'esterno tramite ***provider***. Va precisato che si è scelto di prendere anche i coefficienti beta dall'esterno perché si è voluto operare nell'ottica di un reale utilizzo delle stime del *value at risk* da parte di un *azienda di credito*, la stima deve essere rapida e

allo stesso tempo non richiedere costi elevati di reperimento delle variabili necessarie (costi dovuti a risorse umane o strumenti informatici diversi da quelli preesistenti).
La letteratura non indica un periodo standard da considerare, infatti, da un lato suggerisce di svolgere l'analisi su un ampio arco temporale, per evitare di sovra/sottostimare il rischio influenzati dalle informazioni provenienti da recenti andamenti, dall'altro suggerisce di utilizzare un breve periodo storico in quanto dati troppo vecchi rischiano di falsare la realtà, le stime sarebbero influenzate da shock ormai passati e allo stesso tempo risponderebbero in modo lento a variazioni improvvise delle condizioni di mercato.
Un buon compromesso può vedersi nell'utilizzo di pesi, ponderare in maniera decrescente i dati storici (attribuendo ai più vecchi pesi inferiori) può consentire di ampliare l'arco temporale senza che avvenimenti remoti possano costituire elementi di forte discriminazione, allo stesso tempo i cambiamenti recenti sono presto recepiti dai sistemi di stima ma gli effetti di shock di brevissimo periodo sono attenuati dalla storia.
La scelta del numero di deviazioni standard da considerare, come quella del periodo storico, dipende dalle politiche di gestione del rischio che l'utilizzatore del modello intende attuare, dalla sua propensione al rischio ancorché dalla destinazione del portafoglio, negoziazione o immobilizzazione.
Nella fase empirica noi abbiamo utilizzato solo due intervalli di confidenza pari a 1,65 e corrispondenti rispettivamente al 95 e 99 percento) che sono quelli suggeriti dai modelli e dalla teoria in genere e allo stesso tempo i valori più usati nella fasi applicative.

RISKMETRICS

Il modello *RiskMetrics*, come è stato già detto nel paragrafo precedente, basa la sua specificazione su un'ipotesi fondamentale, la distribuzione normale dei rendimenti semplici, e sull'esistenza di una relazione tra i rendimenti delle azioni e quelli di mercato: è possibile esprimere la variabilità dei rendimenti delle azioni attraverso la volatilità di un indice di mercato.
La formalizzazione del modello prescinde dal rischio specifico di un titolo e considera solo quello derivante dal mercato, detto anche rischio sistematico, poiché si suppone che quelli specifici degli strumenti finanziari che compongono il portafoglio si compensino, in altre parole che la media degli stessi sia uguale a zero.
L'algoritmo utilizzato per il computo del VaR è di tipo lineare ed è di seguito espresso formalmente

$$VaR = \sigma_M \times 1.65 \times \sum_{i=1}^{n} V_i \times \beta_i$$

Dove σ_M è la deviazione standard di un appropriato indice di mercato calcolata su un determinato periodo storico, 1.65 è l'intervallo di confidenza che identifica la probabilità con cui ci si intende coprire nella stima del valore a rischio, V_i è l'ammontare del titolo i-esimo che confluisce nel

portafoglio, β_i è il coefficiente che esprime la variabilità, calcolata su un prescelto periodo storico, dei rendimenti del titolo i-esimo rispetto ai rendimenti di un indice di mercato.
RiskMetrics, come si può notare, ha vantaggi di tipo operativo derivanti dalla semplicità della metodologia e dal numero limitato d'informazioni di dati necessari per le applicazioni, inoltre tali variabili possono essere sia calcolate dall'utilizzatore della procedura che reperibili nell'ambiente esterno.
È certo che le ipotesi sottostanti sono molto forti e possono mettere a rischio la bontà del modello. In particolare va ricordato che la stima così ottenuta prescinde dal rischio specifico delle azioni ed è basata unicamente sul rischio di mercato, ciò può essere valido per portafogli fortemente diversificati e composti da un elevato numero di strumenti finanziari, inoltre il modello prescinde dalla ***correlazione esistente tra i diversi titoli azionari*** e considera solo la loro volatilità. Si lavora supponendo l'***omoschedasticità*** dello scarto quadratico medio dei rendimenti di mercato, ovvero si suppone che, per tutto il periodo su cui è effettuata la previsione, la deviazione standard resti la medesima.
Dalla volatilità giornaliera è possibile calcolare la volatilità a n giorni semplicemente moltiplicando la deviazione standard per la radice quadrata di n: $\sigma_{t+n} = \sigma_t * \sqrt{n}$
Da qui ne deriva che è possibile calcolare il value at risk a più giorni semplicemente moltiplicando la formula su indicata per radice di n

MODELLO SECONDO I CRITERI DI BASILEA

La terminologia "*modello Basilea*" identifica il modello che le banche dovrebbero applicare nel rispetto dei criteri proposti nel 1995 dalla Comitato di Basilea per la vigilanza bancaria. Esso non impone l'utilizzo di nessun modello specifico ma lascia liberi gli istituti di credito di sceglierne uno a condizione che soddisfi alcuni criteri quantitativi.
I principi guida impongono che il *valore a rischio* sia calcolato ogni giorno sui rendimenti decadali (ovvero rendimenti giornalieri moltiplicati per la radice di dieci), ad un livello di confidenza unilaterale del 99%, utilizzando un periodo storico di almeno un anno. Le banche, dunque, possono utilizzare qualunque procedura di stima ritengano più appropriata, ad esempio modelli costruiti sulla base delle matrici di varianze/covarianze, delle simulazioni storiche o delle simulazioni Monte Carlo, adoperando anche criteri più limitanti di quelli proposti dal Comitato, presupposto che esse tengano conto dei rischi sostanziali cui sono sottoposte.
Il modello più semplice dal punto di vista operativo e della maneggevolezza pensiamo possa identificarsi con procedure molto simili a quelle suggerite da RisckMetrics, simili in quanto i vincoli relativi al tipo di rendimento, al livello di confidenza e al periodo storico formulano eccezione ad alcune ipotesi sottostanti RisckMetrics.
Il VaR, sulla base di quanto detto è computato nel seguente modo

$$VaR = \sigma_M \times 2.33 \times \sum_{i=1}^{n} V_i \times \beta_i$$

La misura del valore a rischio è sempre determinata in senso probabilistico, il modello in questione considera un ampio intervallo di confidenza in quanto l'obiettivo del Comitato di vigilanza è quello di una stima prudenziale del VaR di istituti bancari e di conseguenza evitare la bancarotta.
Inoltre il valore a rischio previsto con le procedure dettate dal Comitato di vigilanza è poco preciso poiché l'intervallo di confidenza è molto ampio ed in periodi di bassa volatilità il supporto di tale teoria porterebbe a sovrastimare il rischio e quel che è peggio condizionare negativamente la gestione del portafoglio.

STIMA DEL VAR ATTRAVERSO LA MATRICE DELLE VARIANZE/COVARIANZE

Il *valore a rischio* di un portafoglio o può essere stimato utilizzando la *volatilità* di ogni singolo titolo e la *correlazione* esistente tra i rendimenti di tutte le azioni. Il rischio globale di un pacchetto azionario calcolato come la somma dei rischi di ogni singolo strumento finanziario potrebbe comportare una sovrastima del rischio, infatti, il rischio totale, gode dell'effetto diversificazione del portafoglio. Secondo questa teoria la stima del VaR è calcolata come la somma delle esposizioni relative alle singole posizioni e della correlazione delle stesse, dove queste ultime possono assumere valori negativi e quindi ridurre l'esposizione complessiva del portafoglio.
A livello empirico, quanto appena detto, si traduce in un modello basato sul calcolo della matrice delle vazianze/covarianze., dove quest'ultima deve essere calcolata utilizzando i rendimenti storici di tutti i titoli.
La formalizzazione del modello può essere rappresentata dal seguente algoritmo

$$VaR = \delta \times \sqrt{\sum_i \left(\sigma_i \times V_i\right)^2 + \sum_i \sum_j 2 \times \rho \times \left(\sigma_i \times V_i\right) \times \left(\sigma_j \times V_j\right)}$$

Dove δ è l'intervallo di confidenza che si vuole utilizzare per la stima, σ e V rappresentano rispettivamente la deviazione standard e il valore del titolo (i-esimo o j-esimo), ρ la correlazione tra i rendimenti.
Ricordiamo che ρ è una misura standardizzata della covarianza ed è compreso tra -1 e +1

$$-1 \le \rho_{ij} = \frac{COV(i;j)}{\sigma_i \times \sigma_j} \le +1$$

JPMorgan nel testo RiskMetrics suggerisce di utilizzare questo modello basando il computo delle variabili su rendimenti logaritmici, la distribuzione dei quali e considerata normale, inoltre suggerisce di considerare nulla la media dei rendimenti.

Una tecnica alternativa potrebbe essere di stimare le variabili con la trasformazione delle serie storiche attraverso il computo delle media mobile con ponderazione esponenziale, dove i dati più recenti hanno un peso più alto. Una ponderazione di tipo esponenziale consente di non trascurare completamente informazioni passate, ma allo stesso tempo di dare maggior enfasi agli avvenimenti recenti, così che uno shock di mercato in atto influenza prontamente la stima del valore a rischio e allo stesso tempo i suoi effetti diminuiscono esponenzialmente quando lo shock è passato.
La stima della deviazione standard e della covarianza secondo i criteri suggeriti da JPMorgan sono formalizzati nel seguente modo

$$\sigma = \sqrt{(1-\lambda)\times\sum_{t=1}^{T}\lambda^{t-1}\times\left(r_t-\bar{r}\right)^2}$$

$$COV(i;j) = (1-\lambda)\times\sum_{j=1}^{T}\lambda^{j-1}\times\left(r_{1t}-\bar{r_1}\right)\times\left(r_{2t}-\bar{r_2}\right)$$

Dove T è il numero di osservazioni considerato, r sono i rendimenti dei titoli e λ è il decay factor. cioè un parametro compreso tra 1 e -1.
JPMorgan attribuisce al parametro un valore pari 0.94 se i rendimenti sono giornalieri, di 0.97 se i rendimenti sono mensili.
Il modello qui proposto risulta essere più complesso di quelli visti in precedenza e allo stesso tempo meno maneggevole a livello operativo; non solo necessita di un **elevato numero di informazioni** ma tali informazioni sono **difficilmente disponibili** sul mercato. Per computare il VaR attraverso questa procedura è fondamentale in primo luogo calcolare tutti gli input necessari al suo funzionamento, in un secondo momento è possibile giungere alla stima risolvendo l'algoritmo in funzione degli input.

SCHEMA E-R

L'approccio seguito per la progettazione dello schema E-R prevede la *scrittura immediata di relazioni normalizzate*.
Secondo questo approccio si parte dallo schema concettuale (ad esempio dal diagramma entità-associazioni) e lo si traduce in uno schema relazionale equivalente. Si può dimostrare che le relazioni così ottenute risultano in terza forma normale.
A riprova di questo fatto si può osservare che *le anomalie nascono perché in una stessa relazione si vogliono rappresentare più fatti fra doro indipendenti*. Se questa indipendenza viene evidenziata a priori nello schema concettuale si può evitare di riprodurla seguendo le regole di rappresentazione di entità e associazioni.
Questo approccio è generalmente preferibile a quello della normalizzazione per due buoni motivi:

1. dopo il processo di normalizzazione c'è il rischio di pervenire ad uno schema relazionale in cui alcune relazioni non sono immediatamente interpretabili nei confronti della realtà oggetto di analisi
2. la conoscenza della teoria delle basi di dati relazionali relative ai concetti di normalizzazione diviene meno importante

La progettazione logica delle relazioni NON presenta anomalie, anche se è bene considerare che nel contestualizzare lo schema E-R all'attività di una azienda di credito si sono presentate più situazioni in cui è davvero impossibile evitare la ridondanza.
Alcune di queste sono: oltre ad associare uno strumento finanziario a più portafogli, occorre anche associare un portafoglio a molti strumenti finanziari, oppure nel processo di mappatura su più indici, oltre ad associare un indice a più strumenti finanziari, occorre anche associare uno strumento finanziario a più indici di settore.
Il contesto a cui si riferiscono le regole imposte dalle forme normali espone solo la "norma" (e probabilmente è proprio per questa ragione che le regole in questione si chiamano forme normali) ed è perciò la particolare realtà rappresentata da E2E ad ammettere la *de-normalizzazione,* apportando gli opportuni cambiamenti all'applicazione in modo da evitare anomalie ed inconsistenze.
Il database contiene tre tipologie di ***attributi derivabili***

1. Attributi derivabili da attributi della stessa entità; attributi che nella stessa entità o relazione possono essere ottenuti, istanza per istanza, mediante calcolo.
2. attributi derivabili da attributi di altre entità; possono essere ottenuti mediante funzioni di aggregazione applicate ad istanze collegate da una relazione.
3. Attributi derivabili da operazioni di conteggio.

I vantaggi dei *dati derivati* sono in termini di riduzione del numero di accessi necessari per calcolare il dato derivato e in maggior facilità nella formulazione delle interrogazioni, mentre gli svantaggi sono in termini di maggior occupazione di memoria e di necessità di operazioni aggiuntive per mantenere aggiornato il dato derivato.
Nell'analisi delle ridondanze sono stati presi in considerazione con maggior peso gli svantaggi con conseguente distacco dei dati derivati dalle relazioni.
Particolare attenzione è stata posta al rispetto della terza forma normale nell'individuazione delle **dipendenze transitorie** trasferendo l'informazione derivante dai campi secondari che non dipendono dalla chiave primaria della tabella in una nuova tabella da adoperare come tabella di ricerca.
Il principio generale adottato è quello di adoperare un solo campo per collegarsi al maggior numero di informazioni utili incluse nelle tabelle correlate e questo ha portato ad accettare la duplicazione

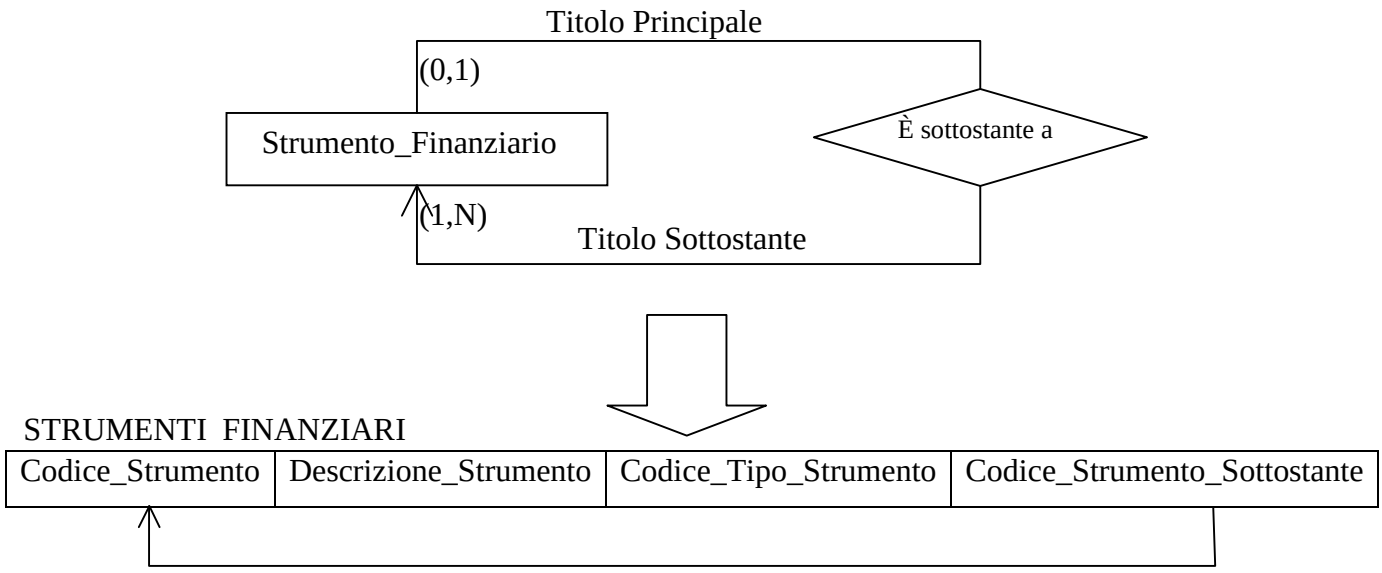
dei valori dei campi di congiunzione (chiavi esterne) giustificandoli come valori di ricerca, qualora ve ne sia bisogno.

Le chiavi utilizzate sono di due tipi:

- FISICA (singola o multipla)
- INTERNA quando nel contesto nessun dato identifica l'elemento (in questi casi si è ricorso a un progressivo o a un codice identificativo interno)

Le chiavi fisiche, oltre ad un ruolo identificativo, portano informazioni sugli elementi (vedi i codici ABI, o gli acronimi utilizzati per identificare le divise o le entità emittente/controparte).

Buona parte della modellazione concettuale dello schema E-R è caratterizzata da associazioni binarie in cui le due entità coincidono, convenzionalmente chiamate **associazioni ricorsive** :



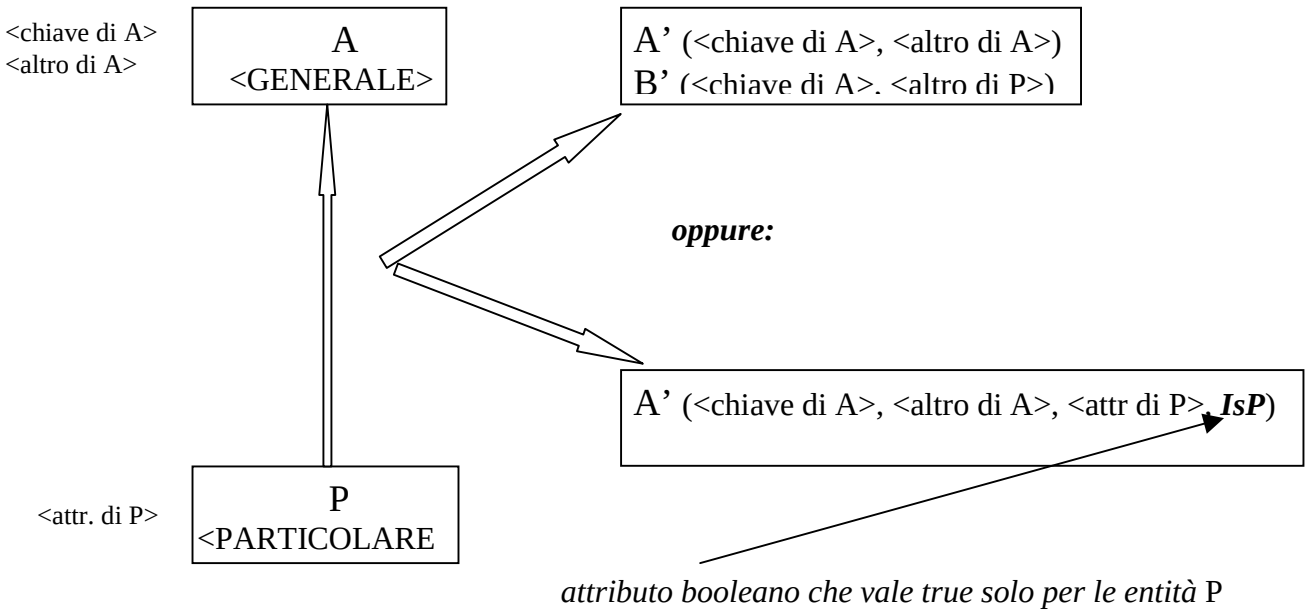
La relazione viene rappresentata con un unico schema logico con possibili valori nulli per la chiave esterna (nome del ruolo dal lato molti).

Da notare che in questo caso l'ordine in cui vengono proposte le coppie assume un ruolo esenziale.

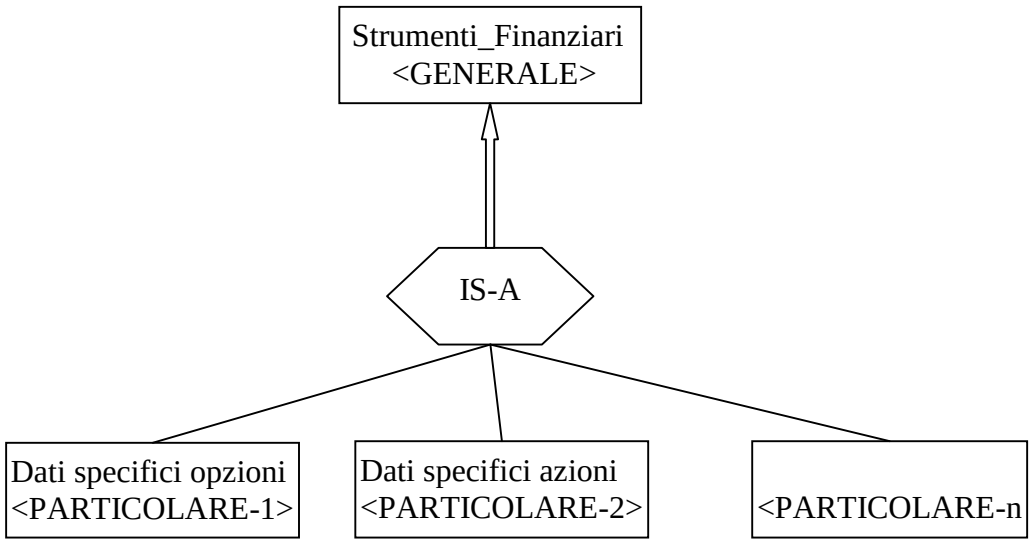
L'inserimento di associazioni ricorsive ha richiesto in fase di interrogazione delle tabelle l'utilizzo di alcune caratteristiche logiche di SQL quali la ridenominazione di tabelle e l'uso di self-join

Associazioni ISA

Il modello ER nella sua versione estesa (*Extended ER* o *EER*) comprende anche strumenti a supporto di quel processo di astrazione noto come generalizzazione. Tali strumenti consentono di classificare gerarchicamente i tipi di entità in modo molto simile a quanto avviene nella programmazione ad oggetti quando si usa il meccanismo di ereditarietà per strutturare le classi.
Lo strumento più semplice è 1' **associazione ISA semplice** (dall'inglese "IS-A", cioè "è un").
In pratica si usa quando si vuole modellare il fatto che nella realtà un certo insieme di entità risulta essere un sottoinsieme di un altro, vale a dire che una certa entità risulta un caso particolare di un'altra più generale. Si dice anche che l'associazione ISA-semplice modella una *gerarchia per sottoinsieme.*
Per quanto detto, le associazioni ISA sono di tipo uno-a-uno, totali rispetto all'entità particolare e parziali rispetto a quella più generale.
A volte c'è la necessità di rappresentare il fatto che diversi tipi di entità concorrono a formare un tipo di entità più generale: questo concetto viene definito **associazione ISA ripartita** (nota anche come *gerarchia per generalizzazione*).
Le associazioni ISA semplici e ripartite possono essere combinate per modellare situazioni gerarchiche che si riscontrano frequentemente nella realtà.
Per quanto riguarda la trasformazione delle ISA, vi sono due soluzioni che si possono prendere in considerazione
Nella prima si rappresenta P con una relazione P' distinta che ha la stessa chiave della relazione che viene associata ad A e gli attributi particolari. Si osservi che la relazione R_P avrà un numero di tuple pari al numero di elementi di P.
La seconda soluzione presenta il vantaggio di richiedere un'unica relazione, ma occorre riconoscere le tuple che rappresentano elementi di P. A tale scopo si propone di introdurre un esplicito attributo booleano ***ISP***. Una soluzione più economica (in spazio) potrebbe essere quella di riconoscere le tuple di P dal fatto che i valori dei suoi attributi sono "nulli" o particolari.

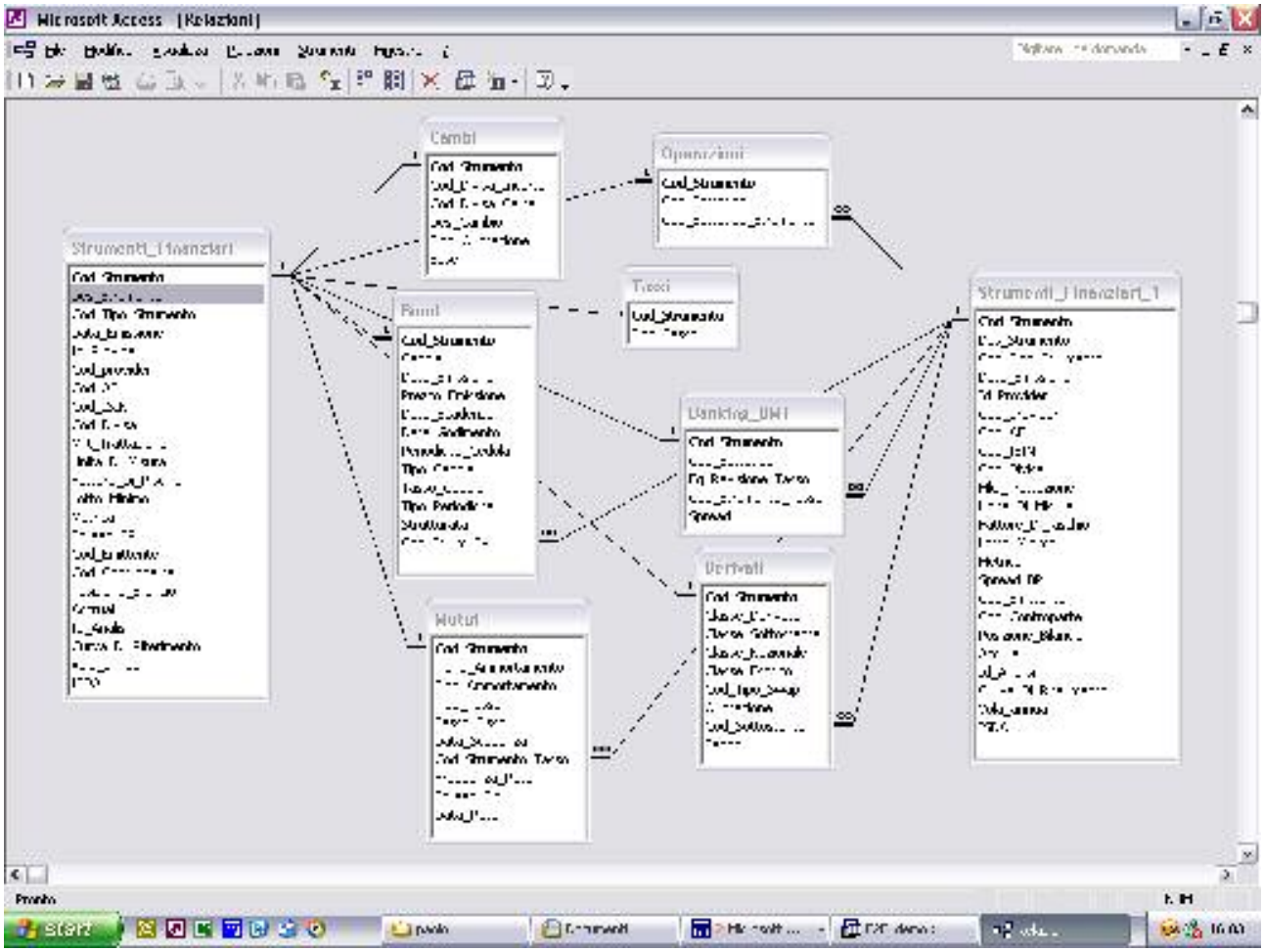


A livello di struttura dati in E2E si utilizza la tecnica **ISA ripartita** volta a rappresentare il fatto che ogni tipologia di strumento finanziario concorre a formare un'entità più generale definita "Strumenti_Finanziari". nella quale sono inseriti gli attributi comuni ad ogni tipologia che risulta quindi essere un caso particolare caratterizzato da propri attributi, cioè particolari proprietà.



La realizzazione adottata la prima delle due soluzione sopra proposte, rappresentando ciascuna associazione ISA con una relazione distinta che ha la stessa chiave della relazione associata all'entità "padre" nominata "Strumenti_Finanziari".

FIGURA 1 : SCHEMI ISA UTILIZZATI



LISTA E DESCRIZIONE TABELLE

TABELLE ADMINISTRATOR

Classe_Derivati : contiene le specifiche della tabella “Derivati”.

- **Classe_Derivato** (Testo): chiave primaria (assume i seguenti valori: Credit Derivative, Forward Rate Agreement, Future, Opzione, Swap); è in relazione uno a molti con la tabella “Derivati”.
- Des_Classe_Derivato (Testo): descrizione

Classi_Opzioni : contiene le specifiche dei derivati di classe “opzione”.

- **Cod_ClasseOPZ** (Contatore): chiave primaria e codice interno; è in relazione uno a molti con la tabella “Specifiche_Opzioni”.
- Classe_Opzione (Testo): contiene tutte le fattispecie di opzioni.
- Des_ClasseOPZ (Testo): descrizione.

Classi_Swap : contiene le specifiche dei derivati di classe “swap”.

- **Cod_Tipo_Swap** (Contatore): chiave primaria e codice interno; è in join con la tabella “Derivati”.
- Codice (Testo): codice parlante; contiene tutte le fattispecie di swap.
- Des_Classe_Swap (Testo): descrizione.
- Note_Classe_Swap (Memo): è possibile inserire descrizioni consistenti.

Classi_Sottostanti : fa parte sempre delle specifiche riguardanti il sottostante dei derivati.

- **Classe_Sottostante** (Testo): chiave primaria e codice parlante; è in relazione uno a molti con la tabella “Derivati”.
- Des_Classe_Sottostante (Testo): descrizione.

Tipi_Fattori_Di_Rischio : è collegata alla tabella “Strumenti_finanziari” e, nell’approccio con ***mappatura a indicatori di rischio*** , indica se lo strumento rappresenta (e con quale modalità) o meno un fattore di rischio.

- **Cod_Fattore_Rischio** (Testo): chiave primaria e codice interno; è in relazione uno a molti con la tabella “Strumenti_Finanziari”.
- Des_Fattore_Rischio (Testo): descrizione.

Tipi_Periodicità_Frequenza : contiene la frequenza temporale.

- **Cod_Periodicita_F** (Contatore): chiave primaria e codice interno; è in relazione uno a molti con le tabelle “Mutui”, “Bond” e “Banking_BMT”.
- Des_Periodicita_F (Testo): descrizione.
- Tempo (Tempo): interpretazione numerica del campo descrizione per gli algoritmi.

Tipi_Posizione_Bilancio :

- **Cod_posizione_ptf** (Testo): chiave parlante (assume valori “attivo”, “passivo”, “port”); è in relazione uno a molti con la tabella “Portafogli” e in join con la tabella “Strumenti_Finanziari”.
- Des_Pos_Bilancio (Testo): descrizione.

Tipi_Quotazioni_Divise :

- **Cod_Tipo** (Testo): chiave parlante; è in relazione uno a molti con la tabella “cambi”.

- Des_Tipo (Testo): descrizione.

Tipi_Ammortamento :

- **Codice_Tipo_Ammortamento** (Contatore): chiave primaria e codice interno; è in relazione uno a molti con la tabella “Mutui”.
- Ammortamento (Testo): descrizione.

Tipo_Interpol : può essere lineare, esponenziale, cubica.

- **Cod_Interpol** (Contatore): chiave primaria e codice interno.
- Tipo_Interpolazione (Testo): descrizione.

Tipo_Metrica : se esiste può essere a lotto minimo o a quantità minima.

- **Cod_Metrica** (Numerico): chiave primaria e codice interno.
- Metrica (Testo): descrizione.

Unita_di_Misura : può essere a valuta, punto indice o punto percentuale.

- **Cod_Unita_Di_Mis** (Contatore): chiave primaria e codice interno; è in relazione uno a molti con la tabella “Strumenti_Finanziari”.
- Des_Unita_Di_Mis (Testo): descrizione.

TABELLE ANAGRAFICHE

Strumenti_Finanziari : è la tabella centrale o proprietaria di tutto il database volta al censimento di tutti gli strumenti finanziari scambiati su mercati regolamentati ed OTC, suddivisi per classi (Banking BMT, Basket, Bond, Cambi, Derivati, Equità, Fund, Index, Merci, Mutui & Finanziamenti, Operazioni, Strutturato, Tasso) ed utilizzati come mattoni principali per ogni funzione di pricing e di valutazione del rischio.
E’ da considerare una tabella cuore, perché ad alta frequenza di modifica e si presenta con un cospicuo numero di righe.
Un ulteriore elemento di distinzione è rappresentato dal suo riutilizzo come copia all’interno dello schema relazionale secondo una logica di seconda istanza/occorrenza che concettualmente ne conferisce il valore di nuova entità all’interno del database con significato e caratteristiche proprie rispetto all’originale (per chiarimenti aggiuntivi vedere la sezione query/join).

- **Cod_Strumento** (Testo): chiave primaria e codice interno. Di seguito riporto le principali relazioni: è in relazione uno a uno con le tabelle “Derivati”, “Bond”, “Mutui”, “Panieri”, Banking_BMT”, “Cambi”, “Operazioni” e “Tassi”; è in

relazione uno a molti con le tabelle “Equity_Fund_Index”, “Componenti_Panieri”, “Dettagli_Strutturati”, “Prezzi”, “Prezzi_Tassi”, “Mappatura_Fr” e “Saldi”.

- Des_Strumento (Testo): descrizione.
- Cod_Tipo_Strumento (Numerico) : è chiave esterna nella relazione con la tabella “Tipo_Strumento”.
- Data_Emissione (Data/Ora) : data emissione strumento.
- Id_Provider (Numerico): è chiave esterna nella relazione con la tabella “Info_Provider”.
- Cod_provider (Testo): contiene il codice associato dal provider di riferimento
- Cod_AF (Testo): codice area finanza.
- Cod_ISIN (Testo): codice bancario interno: vista l’univocità può essere alternativo alla chiave.
- Cod_Divisa (Testo): è chiave esterna nella relazione con la tabella “Divise”.
- Mkt_Trattazione (Testo): è chiave esterna nella relazione con la tabella “Mercati”.
- Unita_Di_Misura (Numerico): è chiave esterna nella relazione con la tabella “Unita_Di_Misura”.
- Fattore_Di_Rischio (Testo): è chiave esterna nella relazione con la tabella “Tipi_Fattori_di_Rischio”.
- Lotto_Minimo (Numerico): Inserimento di quantitativo minimo di acquisto.
- Metrica (Testo): descrizione del lotto minimo.
- Spread_BP (Numerico): Rating Specifico per lo Strumento.
- Cod_Emittente (Numerico): identifica l’entità emittente dello strumento; è legato alla tabella “Entità”.
- Cod_Controparte (Numerico): identifica l’entità controparte dello strumento; è legato alla tabella “Entità” intesa come seconda occorrenza.
- Posizione_Bilancio (Numerico): è legato alla tabella “Tipi_Posizione_Bilancio”.
- Accrual (Numerico): è chiave esterna nella relazione con la tabella “Accrual_Convention” (attenzione non a tutti i tipi strumento).
- Id_Analisi (Numerico): è legato alla tabella “Analisi_Mod”.
- Curva_Di_Riferimento (Testo): descrizione curva di riferimento.
- Vola_annua (Numerico): contiene la misura della volatilità annua dello strumento.

- ISDA (Si/No) : campo relativo ai soli strumenti di classe paniere.

Derivati : è in associazione ISA con la tabella “Strumenti_Finanziari” e rileva da questa i dati specifici degli strumenti derivati.

- **Cod_Strumento** (testo) : chiave primaria e codice interno; è in relazione uno a uno sia con la tabella “Strumenti_Finanziari” che con la tabella “Specifiche_Opzioni”.
- Classe_Derivato (Testo) : i successivi campi si innescano a seconda della tipologia di derivato qui espressa; è chiave esterna nella relazione con la tabella “Classi_derivati”.
- Classe_Sottostante (Testo) : descrizione classe dello strumento sottostante.
- Cod_Sottostante (Testo) : codice strumento sottostante; è chiave esterna nella relazione con la tabella “Strumenti_Finanziari” intesa come seconda occorrenza.
- Classe_Nozionale (Testo) : descrizione (assume i seguenti valori: Irregolare, Regolare).
- Classe_Esotico (Si/No) : descrizione.
- Cod_Tipo_Swap (Numerico) : è legato alla tabella “Classi_Swap”.
- Quotazione (Si/No) : descrizione.
- Segno (Testo) : descrizione (assume i seguenti valori: Buy, Sell).

Specifiche_Opzioni : rileva dalla tabella “Derivati” i dati specifici dei derivati di classe opzione.

- **Cod_Strumento** (testo) : chiave primaria e codice interno; è in relazione uno a uno con la tabella “Derivati”.
- Tipo (Numerico) : è chiave esterna nella relazione con la tabella “Tipo_Opzione”.
- Strike (Testo)
- Scadenza (Data/ora) : Data Scadenza Opzione.
- Classe_Opz (Numerico) : i successivi campi si innescano a seconda della tipologia di opzione qui espressa; è chiave esterna nella relazione con la tabella “Classi_Opzioni”.
- Tipo_Barriera (Numerico) : si innesca solo per le opzioni “con barriera” (nella fattispecie di classe 5)è chiave esterna nella relazione con la tabella “Tipo_Barriera_Opzione”.
- Entita_Barriera (Testo) : descrizione.

- Tipo_Contratto (Numerico) : è chiave esterna nella relazione con la tabella “Tipo_Contratto_Opz”.
- Genere_Opz (numerico) : è chiave esterna nella relazione con la tabella “Genere_Opzione”.

Bond : è in associazione ISA con la tabella “Strumenti_Finanziari” e rileva da questa i dati specifici degli strumenti bond.

- **Cod_Strumento** (Testo) : chiave primaria e codice interno; è in relazione uno a uno con la tabella “Strumenti_Finanziari” e in relazione uno a molti con la tabella “Cash_Flow_Bond”.
- Cedola (Si/No) : Piano_AmmortamentoData_Emissione (Data/ora) : data emissione bond.
- Prezzo_Emissione (Numerico) : prezzo emissione bond.
- Data_Scadenza (Data/Ora) : data scadenza bond.
- Data_Godimento (Data/Ora) : data godimento bond.
- Periodicità_Cedola (Si/No) : ne innesca i campi relativi.
- Tipo_Periodicita (Numerico) : è chiave esterna nella relazione con la tabella “Tipi_Periodicita_Frequenza”.
- Tipo_Cedola (Testo) : descrizione (assume i seguenti valori: Tasso Fisso, Tasso Variabile, Zero Cupon).
- Strutturata (Si/No) : ne innesca i campi relativi.
- Cod_Strum_TV (testo) : codice strumento sottostante; è chiave esterna nella relazione con la tabella “Strumenti_Finanziari” intesa come seconda occorrenza.

Banking_Bmt : è in associazione ISA con la tabella “Strumenti_Finanziari” e rileva da questa i dati specifici inerenti a tutte le operazioni bancarie e interbancarie nel breve-medio termine.

- **Cod_Strumento** (Testo) : chiave primaria e codice interno; è in relazione uno a uno con la tabella “Strumenti_Finanziari”.
- Cod_Sottotipo (Numerico) : è chiave esterna nella relazione con la tabella “Sottotipo_Banking”.
- Fq_Revisione_Tasso (Numerico) : è chiave esterna nella relazione con la tabella “Tipi_Periodicita_Frequenza”.

- Cod_Strumento_Tasso (Testo) : codice strumento sottostante di tipo tasso; è chiave esterna nella relazione con la tabella “Strumenti_Finanziari” intesa come seconda occorrenza.
- Spread (Numerico)

Mutui : è in associazione ISA con la tabella “Strumenti_Finanziari”e rileva da questa i dati specifici inerenti agli strumenti di tipo mutuo.

- **Cod_Strumento** (Testo) : chiave primaria e codice interno; è in relazione uno a uno con la tabella “Strumenti_Finanziari” e in relazione uno a molti con la tabella “Cash_Flow_Mutui”.
- Piano_Ammortamento (Si/No) : ne innesca i campi relativi.
- Tipo_Ammortamento (Numerico) : è chiave esterna nella relazione con la tabella “Tipo_Ammortamento”.
- Tipo_Tasso (Testo) : descrizione assume valori “Fisso” o “Indicizzato” e ne innesca i campi relativi.
- Tasso_Fisso (Numerico) : se a tasso fisso ne esprime l’ammontare.
- Data_Scadenza (Data/Ora) : data scadenza mutuo.
- Cod_Strumento_Tasso (Testo) : se è tasso indicizzato, è il codice dello strumento di tipo tasso indicizzato; è chiave esterna nella relazione con la tabella “Strumenti_Finanziari” intesa come seconda occorrenza.
- Frequenza_Rate (Numerico) : è chiave esterna nella relazione con la tabella “Tipi_Periodicita_Frequenza”.
- Spread_Fin (Numerico)
- Data_Rata (Data_Rata) : la data in cui avviene il pagamento della rata o delle rate (le restanti sono da calcolare poiché si ha la frequenza).

Cambi : è in associazione ISA con la tabella “Strumenti_Finanziari” e rileva da questa i dati specifici inerenti agli strumenti di tipo cambio.

- **Cod_Strumento** (Testo) : chiave primaria e codice interno; è in relazione uno a uno con la tabella “Strumenti_Finanziari”.
- Cod_Divisa_Incerta (Testo) : rappresenta la divisa della moneta a cui è associato il rischio di cambio;è chiave esterna nella relazione con la tabella “Divise”.
- Cod_Divisa_Certa (Testo) : rappresenta la divisa della moneta base; è chiave esterna nella relazione con la tabella “Divise” intesa come seconda occorrenza.
- Des_Cambio (Testo) : descrizione (es.: Cambio EUR/USD).

- Tipo_Quotazione (Testo) : è chiave esterna nella relazione con la tabella "Tipi_Quotazioni_Divise".
- Base (Numerico)

Tassi : è in associazione ISA con la tabella "Strumenti_Finanziari" e rileva da questa i dati specifici inerenti agli strumenti di tipo tasso.

- **Cod_Strumento** (Testo) : chiave primaria e codice interno; è in relazione uno a uno con la tabella "Strumenti_Finanziari" e in relazione uno a molti con la tabella "Dettagli_Curve_Tassi".
- Tipo_Tasso (Numerico) : è chiave esterna nella relazione con la tabella "Tipi_Tassi".

Operazioni : è in associazione ISA con la tabella "Strumenti_Finanziari" e rileva da questa i dati specifici inerenti alle operazioni finanziarie.

- **Cod_Strumento** (Testo) : chiave primaria e codice interno; è in relazione uno a uno con la tabella "Strumenti_Finanziari".
- Cod_Sottotipo (Testo) : è chiave esterna nella relazione con la tabella "Sottotipo_Operazione".
- Cod_Sottotipo_Strumento (Testo) : codice strumento sottostante all'operazione; è chiave esterna nella relazione con la tabella "Strumenti_Finanziari" intesa come seconda occorrenza.

Componenti_Panieri : rileva dalla tabella "Strumenti_Finanziari" i dati specifici inerenti ai panieri intesi come aggregazione di strumenti finanziari.

- **Cod_Paniere** (Testo) : attributo primo e chiave esterna nella relazione con la tabella "Strumenti_Finanziari".
- **Cod_Componente** (Testo) : attributo primo e chiave esterna nella relazione con la tabella "Strumenti_Finanziari" intesa come seconda occorrenza.
- Peso (Numerico) : peso di ciascun componente all'interno del paniere.

Equity_Fund_Index : la tabella aggrega gli strumenti finanziari di classe azione, fondo e indice censiti in "Strumenti_Finanziari" in un'ottica di storicizzazione per data.

- **Cod_Strumento** (Testo) : attributo primo e chiave esterna nella relazione con la tabella "Strumenti_Finanziari".
- **Data_Stacco** (Data/Ora) : attributo primo
- Cedola_Dividendo (Numerico) : importo cedola azionaria.

Dettagli_Strutturati : è simile alla tabella “Componenti_Panieri” : obiettivo di questa tabella è quello di gestire in maniera dinamica la creazione di strumenti finanziari (strutture): prima si creeranno gli strumenti finanziari componenti e successivamente si effettuerà un'aggregazione attraverso questa tabella (ex: Strutturato composto da irs+call+quanto:

- prima si censirà l'IRS con un suo codice etc..
- poi si censirà la Call con un suo codice etc..
- poi si censira la Quanto con il suo codice etc.

infine attraverso la selezione tipo_strumento = struttura si aprirà la tabella "Dett_Struttura" in cui verranno selezionati i componenti della struttura).

- **Cod_Strumento** (Testo) : attributo primo e chiave esterna nella relazione con la tabella “Strumenti_Finanziari”.
- **Cod_Strumento_Componente** (Testo) : attributo primo e chiave esterna nella relazione con la tabella “Strumenti_Finanziari” intesa come seconda occorrenza. Questo campo non fa altro che replicare il precedente e deve essere legato alla tabella Strumenti_Finanziari_(n)
- Quantita (Numerico)

Portafogli : la tabella censisce tutti i tipi di portafogli intesi come aggregazione di strumenti finanziari presenti nella tabella “Strumenti_Finanziari”.

- **Cod_Ptf** (Contatore) : chiave primaria e codice interno; è in relazione uno a molti con le tabelle ”Saldi”, “Storico_Var_Ptf”, “Mappatura_Fr”.
- Cod_Ptf_Utente (Testo) : codice parlante.
- Des_Ptf (Testo) : descrizione
- Cod_Ptf_Predecessore (Numerico) : alcuni portafogli sono composti da portafogli il cui codice è qui inserito; è chiave esterna nella relazione con la tabella “Portafogli” intesa come seconda occorrenza.
- Ptf_Elementare (Si/No) : se il campo è attivo sta ad indicare che il portafoglio in questione può contenere saldi; in altri termini è un portafoglio “foglia” (a differenza dei ptf “padri”che non possono contenere saldi).
- Cod_Owner (Numerico) : identifica l’entità proprietaria del ptf; è chiave esterna nella relazione con la tabella “Entità”.

- Cod_Gestore (Numerico) : identifica l'entità gestore del ptf; è chiave esterna nella relazione con la tabella "Entità" intesa come seconda occorrenza.
- Tipologia (Numerico) : è chiave esterna nella relazione con la tabella "Tipologie_Portafogli".
- Genere (Numerico) : è chiave esterna nella relazione con la tabella "Generi_Portafogli".
- Cod_Posizione_Ptf (Testo) : è chiave esterna nella relazione con la tabella "Tipi_Posizione_Bilancio".
- Flusso_Saldo (Testo) : descrizione saldo.

TABELLE DATI DI MERCATO

Mappatura_FR : la tabella associa agli strumenti finanziari componenti i portafogli un fattore di rischio tramite un parametro appropriato.

- **Cod_Strumento** (Testo) : attributo primo e chiave esterna nella relazione con la tabella "Strumenti_Finanziari"; identifica lo strumento finanziario oggetto della mappatura.
- **Cod_Ptf** (Numerico) : attributo primo e chiave esterna nella relazione con la tabella "Portafogli"; identifica il portafoglio proprietario dello strumento
- **Cod_SF_FR** (Testo) : attributo primo e chiave esterna nella relazione con la tabella "Strumenti_Finanziari" intesa come seconda occorrenza; identifica lo strumento finanziario fattore di rischio.
- Parametro (Numerico) : valore del parametro
- Tipo_Parametro (Numerico) : descrizione parametro utilizzato; è chiave esterna nella relazione con la tabella "Tipologie_Mappatura".
- Data_Calcolo (Data/ora) : data calcolo.
- Cod_Paese (Testo) : descrizione del fattore di rischio.
- Genere (Testo) : descrizione del fattore di rischio.
- Settore (Testo) : descrizione del fattore di rischio.

Saldi : la tabella è finalizzata al reperimento di tutti i titoli posseduti dall'utente con saldo positivo in termini di quantità e facenti o meno parte di un portafoglio.

- Cod_Ptf_Interno (Numerico) : codice numerico indicante il portafoglio a cui si riferisce il saldo (può essere anche blank); è chiave esterna nella relazione con la tabella "Portafogli".

- Cod_Strumento (Testo) : identifica lo strumento a cui il saldo si riferisce; è chiave esterna nella relazione con la tabella “Strumenti_Finanziari”.
- Data_Rif_Saldo (Data/ora) : data di riferimento del saldo
- Saldo (Numerico) : importo saldo
- Tipo_Saldo (Testo) : descrizione saldo (moneta o quantità).
- Divisa_Saldo (Testo) : contiene il codice della divisa di conto se il saldo è monetario
- Cod_Saldo (Contatore) : chiave primaria e codice interno.
- Cod_Ptf_Saldo (Testo) :campo descrittivo.

Prezzi : contiene le serie storiche dei titoli finanziari quotati

- **Cod_Strumento** (Testo) : attributo primo e chiave esterna nella relazione con la tabella “Strumenti_Finanziari”; identifica lo strumento finanziario a cui appartiene la serie storica
- **Data** (Data/ora) : attributo primo; data del prezzo
- Prezzo_Apertura (Numerico) : valore del prezzo di apertura.
- Prezzo_Chiusura (Numerico) : valore del prezzo di chiusura.
- Data_Scarico (Data/ora) : data di scarico del prezzo dal provider.
- Cod_Divisa (testo) : codice della divisa a cui si riferisce il prezzo

Prezzi_Tassi : contiene le serie storiche degli strumenti di tipo tasso

- **Cod_Strumento** (Testo) : attributo primo e chiave esterna nella relazione con la tabella “Strumenti_Finanziari”; identifica lo strumento finanziario tasso a cui appartiene la serie storica.
- **Data** (Data/ora) : attributo primo; data del tasso.
- Data_Scarico (Data/ora) : data di scarico del tasso dal provider.
- Prezzo (Numerico) : ammontare del tasso.
- Divisa (Testo) : codice della divisa a cui si riferisce il prezzo.

Cash_Flow_Bond : contiene i flussi di cassa degli strumenti di tipo bond

- **Cod_Strumento** (Testo) : attributo primo e chiave esterna nella relazione con la tabella “Strumenti_Finanziari”; identifica lo strumento finanziario bond a cui appartiene il flusso.
- **Data_Flusso** (Data/ora) : attributo primo; data di riferimento per i flussi di cassa
- **Data_Fixing** (Data/ora) : data fixing.

- Cod_Strumento_Fin (Testo) :
- Cedola (Numerico) : valore cedola.
- Spread (Numerico) : valore spread.
- Quota_Capitale (Numerico) : valore quota capitale.

Cash_Flow_Mutui : contiene i flussi di cassa degli strumenti di tipo mutuo.

- **Cod_Strumento** (Testo) : attributo primo e chiave esterna nella relazione con la tabella “Strumenti_Finanziari”; identifica lo strumento finanziario mutuo a cui appartiene il flusso.
- **Data_Flusso** (Data/ora) : attributo primo; data di riferimento per i flussi di cassa
- **Data_Fixing** (Data/ora) : data fixing
- Cod_Strumento_Fin (Testo) :
- Quota_Capitale (Numerico) : valore quota capitale.
- Quota_Interessi (Numerico) : valore quota interessi.
- Debito_Residuo (Numerico) : ammontare del debito residuo.

Storico_Mappature : ha funzione di storicizzazione, perciò la data non è più solo un riferimento, ma un campo chiave.

- **Cod_Strumento** (Testo)
- **Cod_Ptf** (Numerico)
- **Data_Strumento** (Data/ora)

STORICO_VAR_PTF : simile alla precedente, ha funzione di storicizzazione.

- **Cod_Ptf** (Numerico)
- **Data_Rif_Calcolo** (Data/ora)
- Valore_Var (Numerico)

RiskMetrics_Frisk : tabella di mappatura degli strumenti finanziari

- **Cod_FR** (testo) : attributo primo; identifica lo strumento finanziario oggetto della mappatura
- Des_FR (Testo) :descrizione
- **Tipo_Fattore** (Testo) : attributo primo; descrizione della tipologia di appartenenza
- Livello (Numerico) :
- Fattore_Decay (Numerico) :

- Volatilita_Prezzo (Numerico) : valore della volatilità dei prezzi
- Volatilita_Rendimento (Numerico) : valore della volatilità dei rendimenti

TABELLE FUNZIONALI AGLI ALGORITMI

Coppie_Prezzi : la tabella è unicamente utilizzata nelle query come deposito temporaneo (tramite query di cancellazione e accodamento) delle serie storiche dei prezzi di due strumenti finanziari filtrati dalla tabella "Prezzi" per fini computazionali.

- **Data** (Data/Ora) : chiave primaria; data del prezzo
- Cod_1 (Testo) : identifica lo strumento finanziario a cui appartiene la serie storica
- Cod_2 (Testo) : identifica il secondo strumento finanziario a cui appartiene la serie storica; può coincidere col primo a seconda delle necessità di calcolo
- Prezzo_1 (Numerico) : valore del prezzo di chiusura del primo strumento
- Prezzo_2 (Numerico) : valore del prezzo di chiusura del secondo strumento
- Ctr (contatore) : progressivo con funzione di supporto alle query

Coppie_Rend : simile alla precedente, è utilizzata nelle query per calcolare e contenere le serie storiche dei rendimenti dei titoli inseriti nella tabella "Coppie_Prezzi".

- **Data** (Data/Ora) : chiave primaria; data del prezzo
- Cod_1 (Testo) : identifica lo strumento finanziario a cui appartiene la serie storica
- Cod_2 (Testo) : identifica il secondo strumento finanziario a cui appartiene la serie storica
- Rend_1 (Numerico) : rendimento del primo strumento
- Rend_2 (Numerico) : rendimento del secondo strumento

Prezzi_Temp : si utilizza per i calcoli delle query

- **Cod_Strumento** (Testo) : attributo primo; identifica lo strumento finanziario a cui appartiene la serie storica
- **Data** (Data/ora) : attributo primo; data del prezzo
- Prezzo_Apertura (Numerico) : valore del prezzo di apertura.
- Prezzo_Chiusura (Numerico) : valore del prezzo di chiusura.
- Data_Scarico (Data/ora) : data di scarico del prezzo dal provider.
- Cod_Divisa (testo) : codice della divisa a cui si riferisce il prezzo
- PRG (Contatore) : progressivo con funzione di supporto alle query

Prezzi_Tassi_Temp : si utilizza per i calcoli delle query

- **Cod_Tasso** (Testo) : chiave primaria; identifica lo strumento finanziario tasso a cui appartiene la serie storica.
- Data_Prezzo (Data/ora) : data del tasso.
- Data_Scarico (Data/ora) : data di scarico del tasso dal provider.
- Prezzo (Numerico) : ammontare del tasso.
- Divisa (Testo) : codice della divisa a cui si riferisce il prezzo.
- PrgTassi (Contatore) : progressivo con funzione di supporto alle query

RiskMetrimcs_MCorr : matrice di correlazione

- **Cod_Mcorr** (Contatore) : chiave primaria
- FR_x (Testo) : codice del fattore di rischio x
- FR_y (Testo) : codice del fattore di rischio y
- Correlazione : correlazione xy

Matrice Covar : è alternativa alla matrice di correlazione

- **Cod_Strumento_1** (Testo) : codice strumento x
- **Cod_Strumento_2** (Testo) : codice strumento y
- Var_Covar (Numerico) : covarianza xy

Coeff_Norm : gestisce i parametri della distribuzione di probabilità normale

- **Alfa** (Numerico) : chiave primaria; il livello di significatività o probabilità di errore
- Cvar (Numerico) : numero di volte che occorre moltiplicare la deviazione standard dello strumento per ottenere l'intervallo di confidenza unilaterale con un livello di confidenza (1-α); in pratica "cvar" è il quantile di livello α/100 di una distribuzione normale standardizzata
- Ces (Numerico) : è il quantile utilizzato nell' Expected Shortfall

Estrazione_Sisba_asiatiche : fornisce i parametri per il pricing delle opzioni (è anomala)

- COD_ABI (Testo) : codice identificativo
- CODISINP (Testo) : codice identificativo
- CURRENT_AVG (Numerico) :
- STRIKE_AGG (Numerico) : prezzo di esercizio (strike price)

- data_elab (Data/ora) : data elaborazione
- tempo (Numerico) : tempo (in frazione di anno)
- coeff_r (Numerico) :
- coeff_s (Numerico) :
- prezzo_spot (Numerico) : prezzo del bene sottostante
- VOLATILITA_A (Numerico) : volatilità del titolo
- rend_spot (Numerico) :
- coeff_opzione (Numerico) :

Risultato_Pricing : riceve i risultati dalla formula di Black e Scholes

- CODISINP (Testo) : codice identificativo
- Pricing (Numerico)
- Delta (Numerico)
- Gamma (Numerico)
- Vega (Numerico)

QUERY

Le seguenti query utilizzano solo codice SQL definito come D.M.L. (data manipulation language) considerata la loro unica funzione di manipolazione dei dati attraverso gli statement SELECT, INSERT, UPDATE, DELETE.

Le clausole utilizzate sono INTO, FROM, WHERE, e il costrutto JOIN.

- La clausola "INTO" definisce i campi del programma usati per ricevere i valori delle colonne ritrovati dallo statement SELECT.
- FROM specifica la tabella/e e le query da cui estrarre le colonne.
- WHERE specifica le condizioni di ricerca che devono essere soddisfatte per la selezione delle righe. Quelle più significative utilizzate in E2E sono:
  - espressioni contenenti i riferimenti ad uno specifico controllo dell'interfaccia grafica utente utilizzati come **parametro**
  - epressioni contenenti **uno o più parametri** combinati con predicati (ad esempio BETWEN), condizioni multiple AND, OR, NOT , funzioni scalari e funzioni IIF per la simulazione di scenari "**if….then**" : soluzione, quest'ultima, utilizzata con particolare successo per i campi delle query in cui è necessario specificare date (o in generale range di valori) da interfaccia utente.

- Il costrutto JOIN per ottenere dati che risiedono su tabelle diverse che possono essere collegati attraverso una colonna comune (colonna di congiunzione) : vengono selezionati record da entrambe le tabelle o query solo se i valori contenuti nei campi collegati con join sono uguali. In E2E sono stati utilizzati prettamente **join interni** e in alcuni casi **join esterni** per far fronte alla necessità che la query selezioni tutti i record da una tabella o da una query a prescindere se abbia o meno dei record corrispondenti nell'altra tabella o query. In particolare E2E è fortemente caratterizzato dal costrutto **self-join** utilizzato sia a livello di **schema E-R** che di **progettazione query** : è possibile collegare con join due copie della stessa tabella o query, creando appunto un **self-join** che combina record della stessa tabella quando vi sono valori corrispondenti nella colonna di congiunzione. Se le tabelle di una query non fossero collegate l'una all'altra tramite self-join (o anche semplice join), direttamente o indirettamente, i record a esse associati non vengono riconosciuti, pertanto vengono visualizzate tutte le combinazioni di record tra le due tabelle. Ad esempio se ciascuna tabella contenesse 10 record, i risultati della query conterrebbero 100 record (10X10). Il set di risultati di tutte le possibili combinazioni viene definito **prodotto cartesiano** o "tra prodotti". Questo tipo di query potrebbe richiedere molto tempo in termini di esecuzione e produrre risultati poco significativi, dunque non sono state inserite in E2E.

In generale, in merito alle query di eliminazione, è possibile che una loro esecuzione elimini record nelle tabelle correlate, anche se queste non sono incluse nella query. Ciò può verificarsi quando la query contiene una tabella che si trova sul lato "uno" di una relazione uno-a-molti e sono state attivate le opzioni di **eliminazione a catena** per quella relazione. Quando si eliminano record dalla tabella "uno", verranno eliminati anche i record dalla tabella "molti".
La situazione appena descritta è solo potenziale all'interno di E2E, considerando che nel corso della progettazione delle relazioni della struttura dati le opzioni di eliminazione a catena non sono state attivate.

Query sottostanti agli algoritmi

Le seguenti query sono innescate secondo una sequenza logica dalla macro "**M_calcolo_vola_corr**" collegata alla maschera "Misure_Volatilita", la cui interfaccia permette la visualizzazione degli indicatori legati alla volatilità degli strumenti finanziari.

Q_canc_coppie_prezzi (query di comando di tipo eliminazione)

Oggetto: tabella "Coppie_prezzi".

Descrizione: rappresenta l'inizio del processo di calcolo della volatilità dei titoli di interesse, nonché la fase preparatoria al processo di selezione e collocazione delle serie storiche finanziarie nella tabella "Coppie_prezzi" (la cui funzione è sopra descritta).

Funzioni: rimozione massiva di tutti i precedenti record dalla tabella "Coppie_prezzi".

Q_crea_coppie_prezzi (query di comando di tipo accodamento)

Oggetto: tabella "Prezzi" e tabella "Prezzi_1". Quest'ultima è la perfetta copia della prima ed è stata inserita nella query, secondo una logica di **seconda istanza**, affinché rappresenti la serie storica di un titolo di seconda scelta diverso da quello filtrato di prima scelta nella tabella originale "Prezzi": anche se fisicamente la tabella presente nel database è una sola, sia a livello concettuale che a livello di compilatore sono due entità distinte dall'uso di un **SELF-JOIN** e dell'**ALIAS** "Prezzi_1", utile quando si accede a più tabelle che hanno colonne con nomi uguali.

I record vengono accodati alla tabella "Coppie_prezzi".

Descrizione: rappresenta la seconda fase del processo; lo scopo della query è di riempire la tabella "Coppie_prezzi" con le due serie storiche degli strumenti di interesse selezionati da interfaccia utente per mezzo dell'utilizzo congiunto di tre filtri:

1. sul codice del primo strumento
2. sul codice del secondo strumento qualora l'utente sia interessato alla correlazione
3. sull'intervallo temporale su cui basare l'analisi

Funzioni: ⟶ la query accoda

- il campo "Cod_Strumento" della tabella "Prezzi" al campo "Cod_1" della tabella "Coppie_prezzi"
- il campo "Cod_Strumento" della tabella "Prezzi_1" al campo "Cod_2" della tabella "Coppie_prezzi".
- il campo "Data" della tabella "Prezzi" al campo "Data" della tabella "Coppie_prezzi".
- il campo "Prezzo_Chiusura" della tabella "Prezzi" al campo "Prezzo_1" della tabella "Coppie_prezzi"
- il campo "Prezzo_Chiusura" della tabella "Prezzi_1" al campo "Pezzo_2" della tabella "Coppie_prezzi"

Ordinamento crescente per i campi "Cod_Strumento" e "Data" della tabella "Prezzi" e "Cod_Strumento" della tabella "Prezzi_1".

⟶ mette a disposizione dell'utente uno strumento versatile per interrogare il database.

In particolare:

- sql “FROM” = “FROM Prezzi INNER JOIN Prezzi AS Prezzi_1 ON Prezzi.Data = Prezzi_1.Data “.
  E’ l’implementazione di un SELF-JOIN reso possibile dall’uso della struttura “AS” per realizzare l’alias “Prezzi_1” e dalla presenza dell’unico (in questo caso) campo “Data” contenente valori corrispondenti nelle due copie cosi da poter essere utilizzato come colonna di congiunzione.

➢ L’efficacia dei tre filtri sopra descritti è garantita da tre condizioni connesse tramite due operatori AND, inserite nella clausola WHERE che ne impone la simultanea verifica ai fini della selezione

1. (Prezzi.Cod_Strumento)=[Forms]![Misure volatilita]![scelta_strumento]) : **attiva il primo filtro applicando come parametro il controllo “scelta_strumento” al campo “Cod_Strumento” della tabella “Prezzi”**

2. (Prezzi_1.Cod_Strumento)=***IIf*** ([Forms]![Misure volatilita]![corr_sino]=True **,** [Forms]![Misure volatilita]![scelta_strumento_2] **,** [Forms]![Misure volatilita]![scelta_strumento]) : **attiva il secondo filtro attraverso l’utilizzo di parametri annidati in una funzione IIF applicata al campo “Cod_Strumento” della tabella “Prezzi_1” per la simulazione del seguente scenario:**

   IF <condizione su correlazione =true> ***THEN***
   <seleziona il valore del controllo “scelta_strumento_2”>
   ELSE
   <seleziona il valore del controllo “scelta_strumento”>
   N.B.: in ogni caso deve essere accodata una seconda serie storica e quindi selezionato un secondo titolo finanziario (anche ripetuto) a fini computazionali.
   N.B.2 : su **“progettazione query**” al carattere “ , “ corrisponde il “ ; ”

3. (Prezzi.Data)<=[Forms]![Misure volatilita]![data_a] And (Prezzi.Data)>=[Forms]![Misure volatilita]![data_da]) : **attiva il terzo filtro** sul campo “Data” della tabella”Prezzi” individuando al suo interno il limite superiore (parametrizzato al controllo “data_a”) e il limite inferiore (parametrizzato al controllo “data_da”) dell’intervallo temporale di interesse : grazie al SELF-JOIN il filtro viene

automaticamente applicato anche al campo “data” della tabella “Prezzi_1” del secondo strumento.

Q_canc_coppie_rend (query di comando di tipo eliminazione)

Oggetto: tabella “Coppie_rend”.

Descrizione: la terza fase del processo di calcolo è simmetrica alla prima; rappresenta inoltre la fase preparatoria al processo di calcolo dei rendimenti sulla tabella “Coppie_rend” (la cui funzione è sopra descritta).

Funzioni: rimozione massiva di tutti i precedenti record dalla tabella “Coppie_rend”.

Q_crea_coppie_rend (query di comando di tipo accodamento)

Oggetto: tabella “Coppie_prezzi” e “Coppie_prezzi_1”; i record vegono accodati alla tabella “Coppie_rend”.

Descrizione: rappresenta la quarta fase del processo; lo scopo della query è di ottenere dalle due serie storiche degli strumenti finanziari già presenti nella tabella “Coppie_prezzi” le serie storiche finanziarie dei rendimenti giornalieri tramite l’espressione degli stessi:

$$r_{t+1} = \frac{p_{t+1} - p_t}{p_t}$$

Dal punto di vista informatico si è adottata una **soluzione ricorsiva** che utilizza un campo contatore (definito “Ctr”) nella tabella “Coppie_prezzi” che consente di selezionare nella medesima solo i record che soddisfano la condizione **contatore=contatore-1** nella tabella immagine “Coppie_prezzi_1” : di conseguenza la selezione su quest’ultima avverrà sui record con prezzo al tempo t+1 mentre sulla tabella “Coppie_prezzi” sarà esattamente al tempo t .

A questo punto vengono calcolati i rendimenti tramite semplici espressioni aritmetiche e poi accodati alla tabella “Coppie_rend”.

Funzioni: ⟶ la query accoda

- il campo “Data” della tabella “Coppie_prezzi” al campo “Data” della tabella “Coppie_rend”.
- il campo “Cod_1” della tabella “Coppie_prezzi” al campo “Cod_1” della tabella “Coppie_rend”.
- il campo “Cod_2” della tabella “Coppie_prezzi” al campo “Cod_2” della tabella “Coppie_rend”.
- il campo calcolato “Rend_1” ottenuto dalla tabella “Coppie_prezzi” al campo “Rend_1” della tabella “Coppie_rend”.

- il campo calcolato “Rend_2” ottenuto dalla tabella “Coppie_prezzi” al campo “Rend_2” della tabella “Coppie_rend”.

⟶ mette a disposizione della query “Q_calcola_vola_corr” le serie storiche finanziarie dei rendimenti degli strumenti precedentemente selezionati.

In particolare :

- sql “SELECT” = SELECT…….(coppie_prezzi_1!prezzo_1-coppie_prezzi!prezzo_1)/coppie_prezzi!prezzo_1 AS Rend_1, (coppie_prezzi_1!pezzo_2-coppie_prezzi!pezzo_2)/coppie_prezzi!pezzo_2 AS Rend_2

Parte dello statement select è adibita allo sviluppo delle espressioni per il calcolo dei rendimenti, il cui risultato verrà assegnato rispettivamente all’alias “Rend_1” e “Rend_2”.

Funzioni di raggruppamento : funzione GROUP BY con associata una funzione HAVING.

In particolare:

- sql “HAVING” = HAVING (((Coppie_prezzi_1.Ctr)=[coppie_prezzi]![ctr]-1)) E’ proprio l’implementazione di una **funzione di raggruppamento ricorsiva** secondo la logica sopra descritta.

Q_calcola_vola_corr (query di selezione)

Oggetto: tabella “Coppie_rend”, “Strumenti_Finanziari e “Strumenti_Finanziari_1”

Descrizione: rappresenta la quinta ed ultima fase del processo di calcolo della **volatilità giornaliera**; la query inoltre è volta alla costruzione di una serie di indicatori quali la **covarianza**, il **coefficiente di correlazione lineare ρ** e il **coefficiente di regressione lineare β** connessi alla volatilità.
Essendo destinata all’interfacciamento con l’utente, ai risultati ottenuti vengono associati alcuni campi descrittivi attinenti gli strumenti oggetto dell’analisi con l’inserimento delle tabelle “Strumenti_Finanziari” e “Strumenti_Finanziari_1” seconda occorrenza in join alla tabella “Coppie_rend”.

Funzioni: ⟶ la query seleziona

- i campi “Cod_Strumento”, “Des_Strumento” e “Id_Provider” della tabella “Strumenti_Finanziari”.
- i campi “Cod_Strumento”, “Des_Strumento” e “Id_Provider” della tabella “Strumenti_Finanziari_1”.

- il campo calcolato “Vol_1” ottenuto come deviazione standard del campo “Rend_1” della tabella “Coppie_rend”.
- il campo calcolato “Vol_2” ottenuto come deviazione standard del campo “Rend_2” della tabella “Coppie_rend”.
- il campo calcolato “r_1” ottenuto come media del campo “Rend_1” della tabella “Coppie_rend”.
- il campo calcolato “r_2” ottenuto come media del campo “Rend_2” della tabella “Coppie_rend”.
- il campo calcolato “r1_r2” ottenuto dall’espressione della media del prodotto dei campi “r_1” e “r_2” della query.
- il campo calcolato “C_Corr” ottenuto dall’espressione del rapporto tra la COVARIANZA (ottenuta come differenza tra il campo “r1_r2” e il prodotto tra i campi “r_1” e “r_2”) e il prodotto tra i campi “Vol_1” e “Vol_2” della query: è il COEFFICIENTE DI CORRELAZIONE LINEARE (o covarianza standardizzata).
- il campo calcolato “C_regr” ottenuto dall’espressione del rapporto tra la covarianza tra i due titoli e la varianza del primo titolo (elevando al quadrato il campo “Vol_1”): è il coefficiente angolare della retta di regressione tra i rendimenti di mercato del titolo_2 su quelli del titolo_1, definito COEFFICIENTE DI REGRESSIONE LINEARE β.
- il campo calcolato “C_tempo” indicante un coefficiente temporale.
- il campo calcolato “tGIORN” indicante l’intervallo temporale preso in considerazione.

NB: La volatilità calcolata su ciascun strumento, viene poi memorizzata attraverso query di aggiornamento sulla tabella “Strumenti_Finanziari”.

Value at Risk (V.a.R.) : algoritmi e query

Tra i diversi metodi per il calcolo del ***valore a rischio*** il prototipo E2E adotta **l’approccio varianza/covarianza** (anche detto approccio parametrico) di cui illustro brevemente le caratteristiche principali.
In questo metodo si assume che le variazioni dei parametri di mercato (tassi di interesse e di cambio, prezzi azionari e delle merci) si **distribuiscano in modo normale** : da ciò deriva che la

media e la varianza dei valori di portafoglio può essere calcolata dalla media e dalla varianza dei parametri di mercato sottostanti.

Per misurare il rischio, la moderna teoria finanziaria deve assumere che la distribuzione di probabilità dei profitti segua una forma normale, e se ciò avviene è possibile utilizzare la deviazione standard quale indicatore del grado di incertezza: **in altri termini è possibile intendere la media come una misura dell'attesa dei profitti futuri e la deviazione standard come un probabile intervallo entro cui cadranno**.

Ad esempio dalla statistica sappiamo che se una variabile si distribuisce normalmente, il 95% delle osservazioni cadrà in un intervallo compreso tra la media più e meno 1.96 deviazioni standard e il 99% delle osservazioni sarà invece compreso in un range pari alla media più e meno 2.58 deviazioni standard, e così via.

Il concetto di Value at Risk si basa su queste considerazioni ed è definibile come una stima, con un certo grado di confidenza, del massimo ammontare di perdita che può verificarsi in un certo periodo di tempo (l'intervallo di confidenza è spesso il 95% o il 99%).

Con il termine "approccio varianza/covarianza" si intende comunemente indicare un metodo di calcolo della deviazione standard del portafoglio, il quale ipotesi più raffinate circa la correlazione esistente tra i beni del portafoglio.

Il V.a.R. di portafoglio è espresso dalla seguente formula:

$$V.a.R. = \sigma_{PORT} * \sqrt{t} * v * N^{-1}(\alpha)$$

$$\sigma^2_{PORT} = \sum_{s=1}^{n} p_s^2 \sigma_s^2 + \sum_{s=1}^{n}\sum_{c=1}^{n} p_s \, p_c \, \sigma_{sc}$$

dove:

σ_s^2 = varianza del singolo asset

σ_{sc} = covarianza tra coppie di asset

p = peso del singolo asset

Ad esempio se si desidera un intervallo di confidenza del 99% allora $N^{-1}(\alpha) = 2.33$ e il numero così ottenuto rappresenta la massima perdita probabile che con il 99% di probabilità non verrà superata nei prossimi t giorni.

Questo metodo tuttavia non è in grado di catturare compiutamente le categorie di rischio di tipo non-lineare come nel caso di portafogli composti ad esempio da opzioni.

In E2E, come descritto più avanti, per la gestione dei rischi di natura opzionale si ricorre ad un **approccio di sensitivity** (come il delta di un'opzione), approccio di tipo deterministico volto a fornire una prima approssimazione del cambiamento di valore di un bene o di un insieme di beni al

modificarsi di un certo parametro di mercato (il prezzo del sottostante nel caso di un'opzione), ma non in grado di incorporare la probabilità che tali modifiche abbiano luogo.
Il VaR, invece, tiene conto di questa probabilità basandosi sulla distribuzione di probabilità empirica (e storica) delle variazioni dei prezzi o dei tassi.
In secondo luogo, le misure di sensitività non tengono conto delle relazioni esistenti tra i diversi beni.

Le seguenti query sono sottostanti al report "**Analisi Rischio**" e collegandosi secondo una sequenza logica d'innesti consecutivi portano al conseguimento del *VaR* e dell'*Expected Shortfall*

Q_calcola_valore_strum (query di selezione)

Oggetto: tabelle "Saldi", "Strumenti_Finanziari" e "Prezzi".

Descrizione: la query è finalizzata al reperimento di tutti i titoli posseduti dall'utente con saldo positivo in termini di quantità (per cui inclusi nella tabella "Saldi") e al calcolo del rispettivo valore.

Funzioni: ⟶ la query seleziona

- i campi "Data" e "Prezzo_Chiusura" dalla tabella "Prezzi"
- i campi "Cod_Strumento", "Cod_Ptf_Interno" e "Saldo" dalla tabella "Saldi"
- il campo "Cod_Tipo_Strumento" dalla tabella "Strumenti_Finanziari" il quale viene poi sottoposto a verifica da una funzione IIF al fine di prezzare su una base diversa gli strumenti di tipo bond quotati a nozionale in euro
- il campo calcolato "Val_strum" ottenuto dall'espressione del prodotto dei campi "Saldo" e "Prezzo" della query.

⟶si innesta nella query "Q_calc_var"

Q_calc_vola_strum (query di selezione)

Oggetto: tabelle "Portafogli", "Mappatura_Fr", "Strumenti_Finanziari", "Strumenti_Finanziari_1", "Tipi_Periodicita_Frequenza", "Coeff _norm".

Descrizione: la query è volta al calcolo del **V.a.R.** e dell'**Expected Shortfall** per ogni singolo strumento all'interno del portafoglio di interesse, secondo un **approccio con mappatura a indicatori di rischio**.

In pratica seleziona e calcola campi atti all'implementazione dell'algoritmo:

$$V.a.R.=\sigma_p * \sqrt{t} * v * N^{-1}(\alpha)$$

Dove $V = \sum_{j=1}^{n} p_j \times P_j$ è la sommatoria dei valore di ogni singolo strumento del ptf già calcolato dalla query "Q_calcola_valore_strum"; $\sigma_p = \sqrt{W' \times \Sigma \times W}$ è la volatilità del portafoglio con w= pesi portafoglio e Σ= MATRICE DI VARIANZA-COVARIANZA di tutti i titoli in portafoglio.
La query sostituisce alla volatilità del singolo strumento il suo Beta moltiplicato per la volatilità del fattore di rischio a cui lo strumento è collegato (nel nostro caso un indice settoriale).

$$\sigma_i = \beta_i \times \sigma_i^*$$
$$\sigma_{ij} = \sigma_{ij}^* \times \beta_i \times \beta_j \quad \text{ottenendo così } \sigma_p = \sqrt{W^* \times \Sigma^* \times W^*}$$

Inoltre il livello di significatività **α** e il quantile associato $N^{-1}(\alpha)$ introdotto dalla query è un parametro fornito da interfaccia utente come pure **l'orizzonte temporale t** su cui faccio il calcolo.

Funzioni: ⟶ la query seleziona

- i campi "Cod_Ptf", "Cod_Ptf_Predecessore", "Cod_Ptf_Utente", "Des_Ptf", "Ptf_Elementare", dalla tabella "Portafogli"
- i campi "Cod_Strumento" e "Cod_SF_FR" dalla tabella "Mappatura_Fr"
- i campi descrittivi "Des_Strumento" dalla tabella "Strumenti_Finanziari" e "Strumenti_Finanziari_1"
- i campi "alfa", "cvar", e "ces" dalla tabella "Coeff_norm"; **il campo "alfa" è sottoposto ad un filtro parametrizzato al controllo "alfa" della maschera "Misure Rischio"**
- i campi "Cod_Periodicita_F" e "Tempo" dalla tabella "Tipi_Periodicita_Frequenza"; **il campo "Cod_Periodicita_F" è sottoposto ad un filtro parametrizzato al controllo "orizzonte" della maschera "Misure Rischio"**
- il campo calcolato "Vol_PTF_Dir"
- il campo calcolato "Vol_PTF_FR" la cui l'espressione "Avg(strumenti_finanziari_1!vola_annua*[parametro])*[tempo]^0.5" restituisce proprio la volatilità $\sigma_i = \beta_i \times \sigma_i^*$ mappata sulla volatilità dell'indice tramite il parametro BETA con riferimento all'unità di tempo di interesse per l'analisi

tramite la radice quadrata del campo “tempo” ; ogni singolo strumento del portafoglio considerato è identificato da un unico record attraverso un raggruppamento multiplo sui campi codice (“Cod_Ptf “,“Cod_Ptf_Predecessore“, “Cod_Ptf_Utente”, “Cod_Strumento”, “Cod_SF_FR”) e a tal fine l’espressione è stata inserita all’interno dell’operatore media che svolge quindi una mera funzione di aggregazione

- il campo calcolato “Var_s” la cui espressione “[vol_PTF_FR]*[cvar]” restituisce il V.a.R. di strumento; **il campo “cvar” rappresenta il numero di volte che occorre moltiplicare la deviazione standard dello strumento per ottenere l’intervallo di confidenza unilaterale con un livello di confidenza (1-α), dove α rappresenta il livello di significatività (o probabilità di errore) espresso in termini percentuali e prefissato dall’utente: in pratica “cvar” è il quantile di livello α/100 di una distribuzione normale standardizzata. Ad esempio, con α=5% l’espressione restituisce perdita minima che ciascun strumento può subire in un giorno nel 5% di casi peggiori**
- il campo calcolato “Es_s” la cui espressione “[vol_PTF_FR]*[Ces]” restituisce l’Expected ShortFall di strumento, simile al V.aR. ma indicante **perdita media che esso può subire in un giorno (o su un periodo storico predeterminato) nel 5% di casi peggiori**

⟶ si innesta nella query “Q_calc_var” e nella query “Q_indicatori_ptf”.

Q_indicatori ptf (query di selezione)

Oggetto: query “Q_calc_vola_strum”

Descrizione: la query estende il calcolo del **V.a.R.** e dell’**Expected Shortfall** dallo strumento al portafoglio di appartenenza

Funzioni: ⟶ la query seleziona

- il campo “Cod_Ptf” e ne esegue il raggruppamento
- il campo calcolato “var_ptf” la cui espressione “Sum([Var_s])*0.835” restituisce il V.a.R. di portafoglio
- il campo calcolato “Es_PTF” la cui espressione “Somma([es_s])*0,841” restituisce l’ Expected Shortfall di portafoglio

⟶ si innesta nella query “Q_calc_var”.

Q_calc_var (query di selezione)

Oggetto: query “Q_calcola_valore_strum”, “Q_calc_vola_strum” e “Q_indicatori ptf” e la tabella “Portafogli”.

Descrizione: scopo della query è quello di unire tramite semplice selezione i risultati ottenuti dall’esecuzione delle tre query precedenti e di renderli disponibili all’interfaccia utente.

Funzioni: ⟶ la query seleziona

- i campi “Cod_Ptf_Interno”, “Cod_Strumento”, “Saldo”, “Val_strum”, “Prezzo” dalla query “Q_calcola_valore_strum”
- i campi “Var_s”, “Es_s”, “Cod_Ptf_Predecessore”, “Des_Ptf”, “Strumenti_Finanziari.Des_Strumento”, “Vol_PTF_Dir”, “Vol_PTF_FR”, “Strumenti_Finanziari_1.Des_Strumento”, “Cod_SF_FR”, “Ptf_Elementare” dalla query “Q_calc_vola_strum”
- i campi “var_ptf” e “Es_PTF” dalla query “Q_indicatori_ptf”
- i campi “Cod_Owner”, “DES_PTF_PRED” dalla tabella “Portafogli”

⟶ rende disponibili i dati al report “Analisi Rischio”.

Approccio delta-gamma e Mathcad application

L’approccio parametrico appena esaminato è adatto per la misurazione dei rischi lineari di tasso, di cambio o di prezzo azionario (in E2E è stato preso in considerazione solo quest’ultimo), mentre per quelli di natura NON-lineare a cui sono soggetti gli strumenti finanziari derivati è preferibile fare ricorso a modelli basati sulla simulazione (storica o di Monte Carlo).
Esiste tuttavia un approccio che introduce un’approssimazione analitica alla relazione di NON-linearità, adottando l’espansione in serie di Taylor.
L’approccio in questione assume che il cambiamento di valore di uno strumento soggetto a dinamiche NON-lineari possa essere approssimato dal **delta** (la derivata prima del valore dell’opzione in relazione allo strumento sottostante) e dal **gamma** (la derivata seconda del valore dell’opzione con riferimento allo strumento sottostante, cioè la derivata prima del delta).
Tale approccio prende anche il nome di approccio **delta-gamma.**
Approcci più generali prendono in considerazione anche le altre greche come il **vega** (la derivata prima rispetto alla volatilità), il **rho** (la derivata prima rispetto ai tassi di interesse) e il **theta** (la derivata prima rispetto al tempo a scadenza).
Il modello applicato in E2E prende in considerazione le prime quattro greche, mentre per la valutazione del valore di mercato la formula di **Black e Scholes** per il calcolo del valore di una **call**

option (opzione che conferisce al possessore la facoltà di comprare un certo strumento finanziario a un prezzo prefissato e a una scadenza predefinita) :

S = prezzo del bene sottostante
σ = volatilità del titolo
r = tasso di interesse
K = prezzo di esercizio (strike price)
t = vita residua in frazione di anno

N(d1) e N(d2) =funzioni di distribuzioni normali cumulative
c = prezzo della call

$$c = S * N\left(d_2\right) - K * e^{-r*t} * N\left(d_1\right)$$

$$d_1 = \ln\left(\frac{S}{K}\right) + \left(r - \frac{\sigma^2}{2}\right) * t$$

$$d_2 = d_1 + \sigma * \sqrt{t}$$

Le equazioni di Black & Scholes sono state sviluppate tramite l'utilizzo di un software matematico: **Mathcad 2001i**.

Il software in questione è caratterizzato dalla facilità d'uso dell'editor nella scrittura degli algoritmi matematici e per essere particolarmente versatile sotto due punti di vista :

- il suo utilizzo è simile a quello di un linguaggio di programmazione, e ciò ha permesso la dichiarazione e assegnazione delle prime cinque variabili sopra elencate

  S := prezzo_spot
  σ := VOLATILITA_A · coeff_s
  r := rend_spot · coeff_r
  K := STRIKE_AGG
  t := tempo

- in secondo luogo si avvantaggia della tecnologia **OLE 2 (object linking and embedding versione 2.0)** che in generale consente alla Microsoft di integrare più applicazioni allo scopo di farle funzionare assieme.

Nel nostro caso si tratta di aggiungere un oggetto creato in un'applicazione mathcad ad un file access (mdb) tramite un processo detto collegamento ed incorporamento di oggetti, ovverosia OLE: si può dunque scegliere se **incorporare** o **collegare** un oggetto da una fonte esterna, considerando che la principale differenza tra un oggetto collegato ed uno incorporato è il luogo in cui sono archiviati i dati.

In E2E si è deciso di incorporare l'oggetto mathcad sulla base delle seguenti motivazioni:

- l'oggetto mathcad è più adatto all'incorporamento perché concepito come **motore di calcolo** perciò NON soggetto ad alcun aggiornamento ma alla necessità di diventare parte del file destinazione

- nessun’altra applicazione ha accesso ai dati contenuti in un oggetto incorporato in un’altra applicazione ma tutti i dati ad esso associati sono ricopiati ed inclusi nel controllo contenitore OLE.

Utilizzo delle interfacce OLE automation: IMathcad interface

Per capire lo scopo e il funzionamento di Imathcad interface vorrei discutere di qualche dettaglio riguardante l’OLE: l’aggiornamento della terminologia dovuto alla necessità di rispecchiare il cambiamento della tecnologia.
La prima implementazione di OLE era progettata per fornire piccoli oggetti riutilizzabili che presentavano un'interfaccia di programmazione e di gestione comune e condivisibile tra applicazioni desktop (es. Ms Office) o anche attraverso una rete locale, e poggiava su di un protocollo preesistente per lo scambio di dati, denominato DDE (Dynamic Data Exchange), che presentava però diverse limitazioni, soprattutto per le sue scarse prestazioni e per il fatto di non essere universalmente applicabile.
E’ l’avvento del **COM (acronimo di Component Object Model, modello ad oggetti dei componenti)** a rappresentare la base ed il nuovo standard per il rilascio dell’OLE 2 e in seguito di nuove tecnologie più avanzate ed innovative come quella dei **controlli ActiveX**.
L'architettura COM è il tentativo effettuato dalla Microsoft di stabilire un comune paradigma per l’interazione tra software di qualsiasi tipo (librerie, applicazioni, software di sistema, ed altro ancora) secondo **modalità standard ed universali** : COM non specifica come debba essere strutturata un’applicazione, i cui dettagli implementativi sono lasciati al programmatore, ma specifica un modello ad oggetti e le condizioni che permettono a tali oggetti, denominati appunto **OGGETTI COM** (detti anche OLE Components), di interagire con altri oggetti.
La maggiore capacità d’interazione è dovuta all'unificazione, mediante un unico standard, delle modalità di comunicazioni tra i vari programmi, utilizzando gli oggetti COM come in un sistema **Client-Server** di rete.
Nulla vieta, infatti, che i processi cliente e servente possano risiedere anche su uno stesso calcolatore dove il canale di comunicazione potrebbe essere virtuale, cioè simulato attraverso la memoria principale.
In genere un client richiede un servizio attraverso una **chiamata di procedura remota** o **RPC** (Remote Procedure Call) per molti aspetti indistinguibile dalle normali chiamate di procedura e la cui sintassi indicativa potrebbe essere :

call nome-servizio(**in** <parametri di ingresso>, **out** <parametri di uscita>)

L'idea alla base dell'architettura COM è, come già detto, quella di permettere a qualsiasi componente software di implementare i propri servizi come oggetti COM.
A questi servizi si può accedere tramite il supporto, da parte degli oggetti stessi, di una o più **interfacce** : "finestre" sul mondo esterno e uniche "vie di accesso" attraverso le quali altre parti software (**Client Applications**) possono inviare o ricevere dati dall'oggetto stesso (**Server**), senza dover necessariamente conoscere il codice del modulo.
In tali interfacce vengono infatti raggruppati dei **metodi** attraverso i quali vengono esplicati tutti i servizi software richiesti.
Un metodo è tipicamente una funzione o una procedura che realizza un'azione specifica; **ogni metodo di un oggetto COM può essere chiamato previo accesso (tramite puntatore) all'interfaccia che lo contiene.**
OLE Automation è una tecnologia legata a OLE 2 e utilizzata dalla versione Mathcad 2001i che permette di inviare i dati dall'applicazione access (come **Automation client**) a mathcad (come **Automation server**) eseguirvi i calcoli e ricevere i risultati in access.
Un programma che utilizza un oggetto di questo tipo è in grado di richiamare qualsiasi funzione messa a disposizione dall'oggetto stesso, anche senza conoscerne a priori l'esistenza (da qui la definizione OLE Automation che, come suggerisce il termine, sta per meccanismo dinamico).
Nel nostro caso Mathcad fornisce due interfacce OLE Automation , **IMathcad** e **IMathcadApplication** : la prima è quella presa in considerazione, i cui quattro metodi permettono proprio il passaggio dati con Mathcad l'esecuzione dei calcoli e l'eventuale salvataggio su file.
La tecnologia Automation è utilizzabile tramite un **linguaggio orientato agli oggetti** in grado di creare e gestire strutture di puntatori e chiamate di funzioni (tramite puntatori) identificate nei metodi come nel nostro caso il **Visual Basic** ; inoltre tramite codice VB definiamo le variabili che andranno a "comunicare" con le ***variabili già definite nell'applicazione Mathcad*** (***variabili Mathcad***) per l'immissione dati e il recupero dei calcoli.
I metodi sono:

- **SetComplex** : assegna il ***numero complesso*** costituito da una ***parte reale*** e una ***immaginaria*** a una variabile Mathcad : sia la parte reale che quella immaginaria possono essere inserite in numero, tramite variabili di tipo variant oppure tramite il riferimento al contenuto di un controllo maschera.
  In E2E si è utilizzato quest'ultimo sistema tranne che per la parte immaginaria inserita tramite una variabile di tipo variant posta =0.
  La sintassi è : longRet = objMC.**SetComplex**(Name As String, RealPart, ImagPart)

- **Recalculate** : forza i calcoli sul documento Mathcad inserito e può essere utilizzato solo dopo la chiamata del metodo SetComplex per assicurare che il contenuto del worksheet Mathcad, compreso l'output, sia coerente con i nuovi input .

  La sintassi è : longRet = objMC.Recalculate
- **GetComplex** : recupera il numero complesso da una variabile Mathcad sempre scomposto nella parte reale e immaginaria che vengono rispettivamente inserite in due variabili di tipo variant : nel nostro caso è il risultato della formula di B&S

  La sintassi è : longRet = objMC.GetComplex(Name As String, RealPart, ImagPart)
- **SaveAs** : è un metodo utile per salvare il contenuto dell'oggetto Mathcad su file.

  La sintassi è : longRet = objMC.**SaveAs**(Name)

Il codice VB completo è il seguente :

```
Private Sub Comando12_Click()
  Dim MathcadObject As Object
  Dim inIm As Variant
  Dim Opricing, Odelta, Ogamma, Ovega, outIm As Variant
Configura l'oggetto Mathcad sul contenitore OLE
  Set MathcadObject = Me.MCAD1.Object

Inizializza a zero le variabili contenenti la parte immaginaria
  inIm = 0
  outIm = 0

Assegna i parametri contenuti nei controlli (in questo caso quelli preceduti dalla locuzione "Me" per intenderci) alla rispettiva variabile Mathcad (il cui nome è racchiuso da apici)
  Call MathcadObject.SetComplex("STRIKE_AGG", Me.STRIKE_AGG, inIm)
  Call MathcadObject.SetComplex("tempo", Me.tempo, inIm)
  Call MathcadObject.SetComplex("coeff_r", Me.coeff_r, inIm)
  Call MathcadObject.SetComplex("coeff_s", Me.coeff_s, inIm)
  Call MathcadObject.SetComplex("prezzo_spot", Me.prezzo_spot, inIm)
  Call MathcadObject.SetComplex("VOLATILITA_A", Me.VOLATILITA_A, inIm)
  Call MathcadObject.SetComplex("rend_spot", Me.rend_spot, inIm)
```

```
Forza i calcoli sul documento Mathcad
  Call MathcadObject.Recalculate

Riceve output da formula di B&S
  Call MathcadObject.GetComplex("BS", Opricing, outIm)
  Call MathcadObject.GetComplex("Delta", Odelta, outIm)
  Call MathcadObject.GetComplex("Gamma", Ogamma, outIm)
  Call MathcadObject.GetComplex("Vega", Ovega, outIm)

Ripone i risultati nei controlli access
  Me.pricing = Opricing
  Me.delta = Odelta
  Me.gamma = Ogamma
  Me.vega = Ovega

End Sub
```

TECNICHE DI VALUTAZIONE DEGLI ALGORITMI DI COMPLESSITA' COMPUTAZIONALE DELLE QUERY

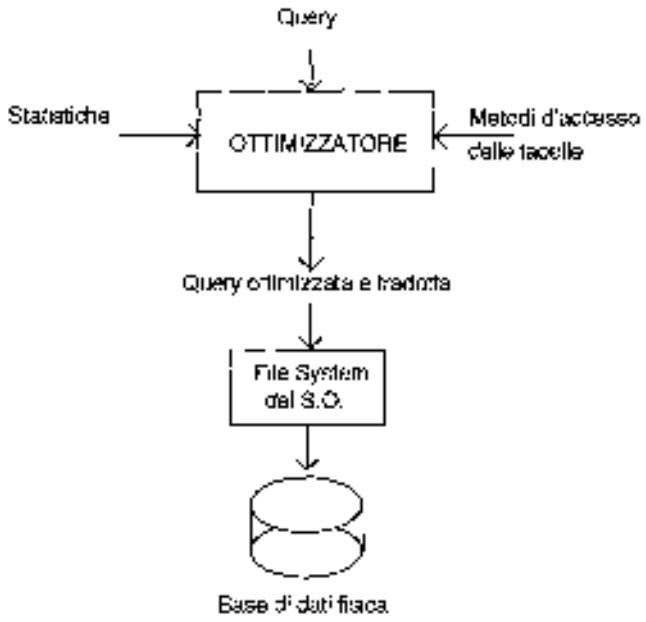
INTRODUZIONE

Prima di proseguire ulteriormente nella trattazione, ritengo opportuno sottolineare come l'esecuzione di algoritmi complessi in E2E abbia comportato anche la realizzazione di molte query (in generale solo stored procedures) e di conseguenza una attenta valutazione delle *performance di calcolo*.

A tal fine, per poter utilizzare in modo efficiente il motore del database è importante sapere non solo come funziona, ma anche conoscere il miglior *meccanismo di ottimizzazione* della base dati compresa l'eventuale denormalizzazione sopra descritta.

I passaggi possono essere diversi per query che NON ritornano risultati (query di comando); concentrerò l'analisi solo sulle query che ritornano dei risultati (*query di interrogazione*) dato che più spesso richiedono ottimizzazione e soprattutto è a queste che si riconduce la complessità degli algoritmi utilizzati in E2E.

Di seguito si affronterà quindi il problema del recupero delle registrazioni che soddisfano un'interrogazione. Questo problema si presenta in tutti i sistemi per la gestione di basi di dati, ma verrà trattato seguendo l'approccio adottato nel progetto dell'ottimizzatore dei sistemi relazionali, cioè di quel modulo del sistema che ha lo scopo di scegliere la strategia ottimale per la sintesi di interrogazioni espresse in modo indipendente dall'organizzazione fisica della base di dati: la trattazione presenta comunque una metodologia generale e utilizzabile per valutare e confrontare strategie di accesso agli archivi.



Inoltre il costo dei possibili piani di accesso, tra cui l'ottimizzatore sceglie quello di costo minimo, è notevolmente influenzato dalla situazione dell'ambiente in cui si deve operare: per questo motivo, si faranno alcune ipotesi sull'ambiente di lavoro per rendere più semplice l'analisi del problema, passando poi all'esame dei principali algoritmi di sintesi delle interrogazioni.

Ottimizzazione delle interrogazioni

Nei modello dei dati gerarchico e in quello reticolare, gli operatori elementari sui dati sono di tipo **procedurale**, **cioè agiscono su un dato alla volta**, ed è compito del programmatore stabilire come essi vadano utilizzati nelle applicazioni per **produrre** il risultato desiderato. Al contrario, nel modello relazionale il risultato desiderato viene specificato con un'espressione dell'algebra relazionale, o in modo **dichiarativo** con una formula del calcolo relazionale, ed è compito dell'*ottimizzatore* del sistema stabilire il **piano di accesso** più adatto per trovare il risultato facendo uso degli operatori e delle strutture dati della macchina astratta fisica.'
In generale esistono diverse strategie alternative per procedere, cosa tanto più vera quanto più un'interrogazione è complessa, e compito dell'ottimizzatore del sistema è di trovare la strategia migliore. Tipicamente l'ottimizzatore esplora in modo controllato lo spazio delle possibili soluzioni scegliendo quella di costo inferiore. Il problema rientra nella categoria dei problemi "difficili" e quindi di solito si usano procedure euristiche con l'obiettivo di trovare rapidamente una buona soluzione. La complessità del problema è dovuta al fatto che (a) un'interrogazione può essere trasformata in più modi in un'altra equivalente, (b) gli operatori dell'algebra possono essere realizzati in più modi e (c) possono esistere più strutture che agevolano l'accesso ai dati.
L'esempio che segue mostra come anche un'interrogazione relativamente semplice si presti a esecuzioni alternative con costi molto diversi.

Esempio

Date due relazioni *Studenti (Matricola, Nome, Provincia, AltreInformazioni)* e *Esami(Materia, MatricolaCandidato, Voto, Data),* si vogliono trovare i dati che soddisfano l'interrogazione "trovare il nome e la data degli esami degli studenti che hanno superato l'esame di DA con trenta", espressa in SQL come:

select Nome, Data
from Studenti, Esami
where *Matricola Candidato = Matricola*
AND Materia = "DA" AND Voto = 30

Si supponga che esistano 2000 studenti e 20000 esami, di cui però solo 500 riguardino l'esame "DA", e di questi solo 50 siano stati superati con 30. Inoltre, per semplificare l'analisi, si supponga

che il tempo di esecuzione di una strategia sia proporzionale al numero di ennuple lette o scritte, e che l'unica possibilità di accesso sia la scansione sequenziale delle relazioni. Per generare la risposta l'ottimizzatore potrebbe procedere nei modi seguenti:

1. **for** s **in** *Studenti* **do**
for e in *Esami* **do**
memorizza (s, *e)* in T;
for *t* **in** *T* **with** *Matricola Candidato* = *Matricola* ... **do**
print(*Nome*, *Data*);

calcola il prodotto cartesiano delle due relazioni, producendo una relazione temporanea di 40000000 ennuple, con un tempo di esecuzione proporzionale a 80000000. Da questa relazione estrae le 50 ennuple che soddisfano la condizione di restrizione e proietta il risultato sugli attributi desiderati. Il tempo di esecuzione di questa strategia è proporzionale a 120000000;

2. **for** e **in** *Esami* **with** *Materia* = "DA" AND *Voto* = 30 **do**
memorizza *e* in T;
for s in *Studenti* **do**
for *t in T* with *Matricola Candidato* = *Matricola* **do**
print(*Nome*, *Data*);

restringe la relazione *Esami* alle ennuple dell'esame "DA" superati con voto 30, producendo una relazione temporanea di 50 ennuple, con un tempo di esecuzione proporzionale a 20050.
Poi fa la giunzione della relazione *Studenti* con la relazione temporanea e proietta il risultato sugli attributi desiderati, con un costo proporzionale a 100000. Il costo totale di esecuzione di questa strategia è proporzionale a 120050, inferiore al costo della strategia precedente di circa un fattore 1000.
Anche se nella realtà le strategie per eseguire un'interrogazione sono più di quelle appena viste, dovendo tener conto anche delle strutture di accesso che si possono impiegare, e che il parametro di costo dipende principalmente dal numero di pagine della memoria permanente trasferite, l'esempio precedente è sufficiente per fornire un'idea dell'importanza dell'ottimizzazione delle interrogazioni al fine di migliorare le prestazioni di un sistema relazionale; ad esempio, interventi sui criteri di preparazione dei piani di accesso per il sistema DB2 hanno portato a un miglioramento delle prestazioni di circa il 60%su alcuni tipi di interrogazioni, passando dalla prima alla seconda versione del sistema.

L'ottimizzazione può essere in generale **dinamica** o **statica**.
Nel caso di accesso interattivo alla base di dati, l'ottimizzazione è sempre dinamica, cioè il piano di accesso è generato al momento di eseguire l'interrogazione tenendo conto dei valori delle statistiche e della presenza di strutture di accesso che in quel momento sono registrate nei cataloghi del sistema.
Nel caso di interrogazioni immerse in un linguaggio di programmazione l'ottimizzazione può essere sia dinamica che statica. Nel primo caso il piano di accesso è generato al momento dell'esecuzione dell'interrogazione, pertanto l'ottimizzazione è fatta ogni volta che viene eseguita l'interrogazione, in base alle grandezze attuali in gioco. È la soluzione usata ad esempio dal sistema Oracle. Poiché in generale l'ottimizzazione è una operazione abbastanza dispendiosa in termini di tempo di calcolo, con l'ottimizzazione statica, usata ad esempio nel sistema DB2, il piano di accesso è generato al momento dell'analisi dell'interrogazione da parte del precompilatore. Pertanto l'ottimizzazione è fatta una sola volta, indipendentemente dal numero di volte che verrà eseguita l'interrogazione, in base alle stime delle grandezze in gioco valide al momento della precompilazione. Una modifica dell'organizzazione fisica dei dati, ad esempio per aggiunta o rimozione di indici, rende non validi certi piani di accesso e comporta una loro rigenerazione riapplicando l'ottimizzazione.

Ipotesi di lavoro

Si studierà il problema dell'ottimizzazione delle interrogazioni sotto certe ipotesi restrittive riguardanti il *tipo di interrogazioni*, la *granularità dell'ottimizzazione, l'organizzazione fisica dei dati, il modello dei costi*, la *gestione del buffer* e le *statistiche* a disposizione.

Tipo di interrogazioni

Le interrogazioni sono espresse nel seguente sottoinsieme del linguaggio SQL:
select *ListaDiAttributi*
from *ListaDiRelazioni*
where *Condizione*
dove:
- *ListaDiAttributi* sono gli attributi che interessano nel risultato (* sta per tutti gli attributi);
- *ListaDiRelazioni* sono le relazioni coinvolte; per semplicità ci limiteremo al caso di al più due relazioni;
- *Condizione* è un prodotto logico di somme logiche di condizioni semplici (detti anche *predicati)* sugli attributi *Ai* delle relazioni. In particolare si supporrà che tali condizioni semplici siano del tipo:
  Ai **Op** vi, con **Op** $\in \{\rangle, \geq, \langle, \leq, =, \neq\}$
  Ai **in** (v_i; ... ; v_N)

Ai **between** v_1 **and** v_2

Ai = Aj (*condizione di giunzione*)

dove attributi Ai identici di relazioni diverse Rs e Rt si rendono diversi ponendo Rs.Ai e Rt.Ai.

Quando la condizione coinvolge due relazioni R ed S, si supporrà che essa sia nella forma: ψ_R AND ψ_S AND , dove ψ_{Join} è una condizione sugli attributi di R, ψ_S è una condizione sugli attributi di S e ψ_{Join} è una condizione di giunzione.

Si definisce **predicato di ricerca**, o **predicato risolubile**, un predicato per il quale *(a)* esiste un indice utilizzabile per trovare le registrazioni che soddisfano il predicato e *(b)* l'indice è utile per ridurre il costo della verifica della condizione. Ad esempio, nel caso di condizioni del tipo $A_1 + A_2 = 1000$, oppure $A_2 \neq 50$, con A_1 e A_2 attributi con indici, le condizioni non sono un predicato di ricerca perché gli indici non aiutano a limitare le ennuple da visitare per verificare la condizione. Un predicato non di ricerca viene detto **predicato residuo**.

Pertanto i predicati di ricerca sono del tipo:

AttributoConIndice **Op** v_i, con **Op** $\in \{\rangle, \geq, \langle, \leq, =\}$

Attributo ConIndice **in** (v_i; ... ; v_N)

AttributoConIndice **between** v_1 **and** v_2

PredicatoDiGiunzione, con almeno un attributo con indice.

Si definisce **fattore booleano** un predicato che, se falso, rende falsa tutta la condizione; tutte le registrazioni del risultato, quindi, soddisfano i fattori booleani

Esempio

Si consideri la relazione Impiegati (Codice, Nome, Qualifica, Stipendio, Città), con indici su tutti gli attributi, e l'interrogazione

select Nome, Stipendio
from Impiegati
where Qualifica= "Programmatore"
AND (Città = "Bologna" OR Stipendio > 2.000.000)

I fattori booleani sono due, ma solo il primo è un predicato risolubile con l'indice su Qualifica.

Granularità dell'ottimizzazione

L'ottimizzazione è fatta separatamente per ogni interrogazione, come accade in tutti i sistemi commerciali; pertanto è bene formulare interrogazioni complesse e non scomporle in tante interrogazioni più semplici perché ciò fa perdere i vantaggi dell'ottimizzazione.

Organizzazione fisica dei dati

I dati sono organizzati in insiemi omogenei di registrazioni con attributi elementari, denominati relazioni di ennuple. In generale, salvo diverse specifiche, si supporrà che una pagina della memoria permanente contenga solo dati di una relazione e una relazione sia tutta contenuta in un solo archivio. In alcuni sistemi invece, come DB2 e SQL/DS, un archivio può contenere più relazioni e una pagina può contenere ennuple di relazioni diverse.

Nel recupero dei dati soddisfacenti un'interrogazione si utilizza la scansione sequenziale o si utilizzano indici costruiti su singoli attributi.

Gli indici per chiave primaria sono densi e organizzati a B+-alberi; nel caso di chiavi secondarie gli indici sono a liste invertite sempre strutturati come B+-alberi, **con foglie contenenti i valori dell'attributo seguiti ciascuno dall'insieme ordinato degli identificatori interni,** o **TID**, delle ennuple che hanno quel valore dell'attributo. In alcune realizzazioni, invece, l'insieme degli identificatori non viene tenuto ordinato.

Si distinguono poi due tipi di indici, per chiave primaria o secondaria, detti di ***ordinamento*** o ***non di ordinamento.***

Un indice di ordinamento (*clustered*) è un indice costruito sull'attributo di ordinamento della relazione. Ciò significa che l'accesso ai dati per mezzo dei riferimenti nell'indice recuperati con una scansione dei nodi terminali, ad esempio per valori crescenti della chiave su cui è costruito tale indice, **non comporta accessi ripetuti** a una stessa pagina dei dati. **Una relazione può essere ordinata su un solo attributo e può avere quindi un solo indice di ordinamento.**

Un indice non di ordinamento (*unclustered*) è un indice costruito su un attributo non di ordinamento della relazione. A differenza del caso precedente, l'accesso ai dati per valori crescenti della chiave provoca **accessi casuali alle pagine della relazione**, che possono essere visitate più volte in momenti diversi. I valori della chiave nelle foglie sono anche essi ordinati, ma la lista di TID associata a ogni valore di chiave può essere mantenuta ordinata o disordinata. Se è ordinata la ricerca di tutte le ennuple aventi un certo valore di chiave comporta la visita delle ennuple nello stesso ordine in cui sono state memorizzate e nessuna pagina della relazione viene letta due volte.

La Figura 2 mostra un esempio dei tre tipi di indice costruiti su una relazione di due attributi contenente dodici ennuple e ordinata in base ai valori del primo attributo A_1. Le ennuple racchiuse dentro un riquadro si suppone che siano memorizzate nella stessa pagina. L'indice I_1 sull'attributo A_1 è di ordinamento, mentre gli indici I_2 e I_3 sull'attributo A_2, rispettivamente con liste di TID ordinate e disordinate, sono non di ordinamento. Nella ricerca di ennuple che soddisfano a condizioni del tipo $A_2 < v_i$, usando l'indice I_2 può accadere che la stessa pagina sia esaminata più volte

(ad es. $A_2 < Y$) e ciò può accadere anche nel caso di condizioni del tipo $A_2 = v_i$ usando l'indice I_3 essendo le liste dei TID non ordinate.

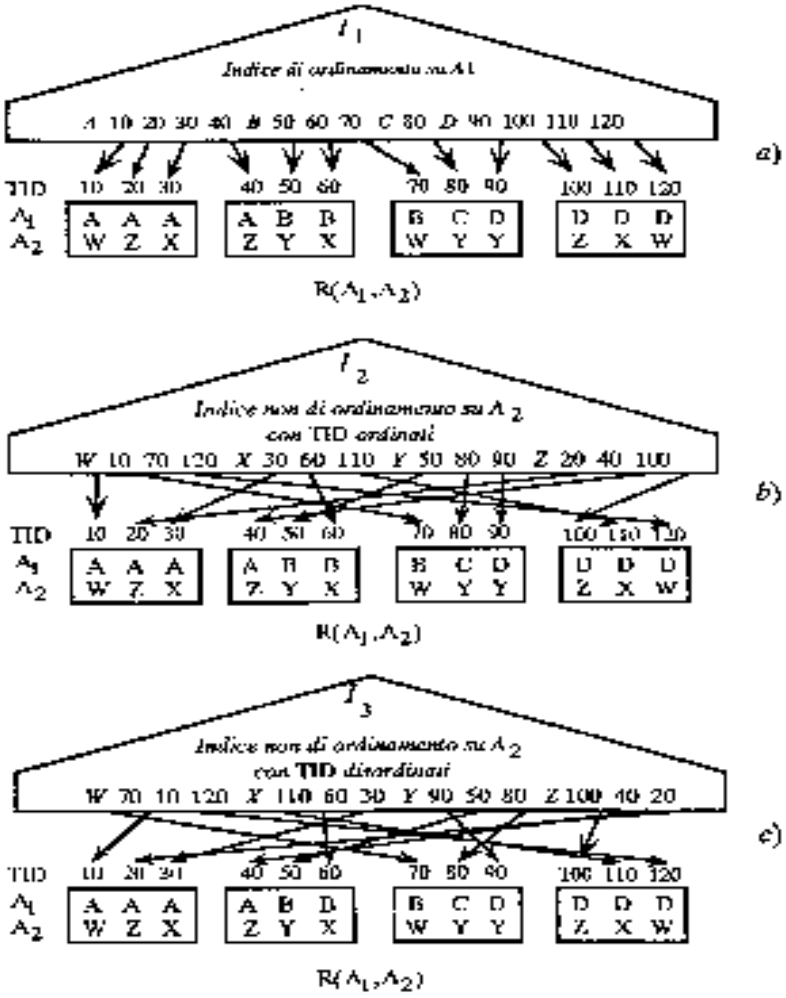


FIGURA 2

Modello dei costi

II costo di esecuzione C_A di un piano d'accesso è espresso come una combinazione lineare del numero di accessi alla memoria permanente, N_A, e il numero di registrazioni N da elaborare in memoria temporanea per produrre il risultato finale:

$$C_A = N_A + fc*N$$

Il fattore fc è un parametro specifico di ogni sistema di gestione di basi di dati; a titolo di esempio, per il *System R* (il prototipo dei sistemi DB2 e SQL/DS), il suo valore poteva essere fissato dall'amministratore della base di dati nell'intervallo da 1/10 a 1/3. Spesso il costo delle operazioni in memoria temporanea è ignorato e per questa ragione nel seguito si usa l'approssimazione

$$C_A = N_A$$

In generale, il costo N_A dipende da come si visitano le ennuple.

Nel caso di scansione sequenziale della relazione R, CA = N_{pag}(R).

Nel caso di accesso con indici, CA è costituito da due termini:

$$C_A = C_I + C_D$$

dove, C_I è il costo di accesso alle pagine degli indici per individuare i TID delle ennuple che soddisfano la condizione; mentre C_D è il costo di accesso alle pagine contenenti i dati.

Il costo C_I si approssima di solito con il costo di accesso ai nodi terminali (foglie), trascurando il costo della visita del cammino dalla radice a un nodo terminale.

Gestione del buffer

E' sufficiente assumere che il gestore del buffer allochi una pagina per ogni relazione coinvolta nell'interrogazione e una pagina per una foglia di ogni indice utilizzato dal piano di accesso. Pertanto ogni volta che viene richiesta un'ennupla (o un TID) che non si trova nella pagina presente nel buffer occorre leggere una nuova pagina (page fault). Sotto queste ipotesi, che generalmente comportano stime in eccesso, il costo di esecuzione del piano di accesso è stimato come il numero di volte che la ricerca dell'ennupla (o del TID) corrente viene effettuata su una pagina diversa da quella che contiene la ennupla (o il TID) trovata immediatamente prima.

Statistiche

I valori di ogni attributo si suppongono uniformemente distribuiti nelle ennuple di una relazione e sul tipo su cui sono definiti.

Per quanto riguarda la distribuzione delle ennuple con un dato valore di un attributo, nelle pagine contenenti la relazione, si distinguono due casi:

- la relazione è ordinata su quell'attributo e, quindi, le ennuple con lo stesso valore dell'attributo sono fisicamente adiacenti;
- la relazione non è ordinata su quell'attributo e le ennuple con lo stesso valore dell'attributo sono uniformemente distribuite nelle pagine contenenti la relazione.

Si assume inoltre che la distribuzione dei valori di un attributo nelle ennuple della relazione sia indipendente dalla distribuzione dei valori di un altro attributo della relazione (uniforme distribuzione e indipendenza tra i valori di attributi distinti). Si supporrà infine che siano note le seguenti statistiche:

1. Per ogni relazione R(A_1, A_2, ... A_n):
    (a) N_{reg}(R), il numero delle registrazioni di R;
    (b) N_{pag} (R), il numero delle pagine occupate da R;
    (c) N_a(R), il numero di attributi di R;
    (d) L_r(R), la dimensione (numero di byte) di un'ennupla di R.
2. Per ogni attributo A_i:

(a) *card*(A_i), il numero di valori distinti del dominio di definizione di A_i;

(b) $L_k(A_i)$, la dimensione di un valore di A_i;

(c) *min*(A_i) e *max*(A_i), il minimo e massimo valore di A_i, se definito su un dominio numerico.

3. Per ogni indice su un attributo A_i:

(a) $N_{key}(A_i) = N_{reg}(\pi\ A_i\ (R))$, il numero di valori distinti di A_i presenti nell'indice. $N_{key}(A_i) = card(A_i)$ quando ogni possibile valore di A_i è presente in almeno un'ennupla;

(b) $N_{leaf}(A_i)$, il numero di foglie dell'indice, stimato con l'espressione:

$$N_{leaf}(A_i) = \frac{N_{reg}(R) * L_{TID} + N_{key}(A_i) * L_k(A_i)}{D_{pag} * f_r}$$

dove L_{TID} è la dimensione dei TID, f_r ($0 < f_r \leq 1$) il fattore di riempimento dei nodi, D_{pag} è la dimensione delle pagine.

Queste statistiche costituiscono i parametri fondamentali su cui si basano le formule dei costi dei piani di accesso ai dati. Nei sistemi relazionali esse vengono mantenute in un catalogo dei dati, in modo da essere utilizzabili dall'ottimizzatore per la scelta della migliore strategia d'accesso. Misure effettuate su alcuni sistemi relazionali hanno anche mostrato che in generale le ipotesi precedenti portano a risultati soddisfacenti.

In particolare, le assunzioni sull'uniforme distribuzione dei valori sono giustificate dal fatto che il rilevamento e la manutenzione di statistiche più significative comportano operazioni di un certo costo, ad eccezione di quei casi rappresentabili con distribuzioni ben note.

Sono stati tuttavia proposti alcuni metodi per la sintesi delle interrogazioni che tengono conto in maniera più dettagliata delle effettive distribuzioni e delle correlazioni tra campi distinti.

I valori memorizzati nel catalogo sono delle stime o possono essere valutati periodicamente sul sistema funzionante con un opportuno comando. Ad esempio, nel sistema DB2, il caricamento delle relazioni e la creazione degli indici inizializzano i valori dei parametri usati dall'ottimizzatore, che possono essere aggiornati con il comando UPDATE STATISTICS ON *Relazione*.

L'aggiornamento di questi valori potrebbe essere fatto automaticamente dopo ogni operazione di inserzione, eliminazione o modifica, ma ciò provocherebbe una serie di operazioni aggiuntive sulla base di dati e sul catalogo, con probabile degradazione delle prestazioni per le operazioni in corso.

Fasi del processo di ottimizzazione

II processo di ottimizzazione in generale è complesso per i numerosi parametri da prendere in considerazione. Per renderlo più rapido, in modo che il tempo per l'ottimizzazione incida poco sul

tempo di esecuzione dell'interrogazione, si utilizzano procedure euristiche per ridurre lo spazio delle soluzioni possibili da esaminare e si scompone il processo in fasi, in ognuna delle quali vengono affrontati separatamente alcuni aspetti del problema, senza mai riprendere in esame le decisioni prese in una fase precedente. Una possibile decomposizione in fasi del processo di ottimizzazione è la seguente: **analisi e semplificazione dell'interrogazione**, **ottimizzazione algebrica** e **ottimizzazione fisica.**

Nella fase di analisi e semplificazione dell'interrogazione, vengono eseguite azioni del tipo:

1. analisi della correttezza lessicale, sintattica e semantica dell'interrogazione, tenendo conto della definizione della base di dati;
2. se l'interrogazione fa riferimento a una tabella definita come risultato di un'altra interrogazione (view), sostituzione dei riferimenti alla tabella con l'espressione dell'algebra che la definisce;
3. generazione di un'opportuna rappresentazione interna, usando gli operatori dell'algebra relazionale e normalizzando la condizione nella forma prodotto di somme logiche;
4. semplificazione della condizione applicando regole di equivalenza di espressioni logiche del tipo:
   (a) ψ_i AND NOT $(\psi_i) \equiv$ *falso*;
   (b) NOT $(\psi_1$ OR $\psi_2) \equiv (\text{NOT}\psi_1)$ AND $(\text{NOT}\psi_2)$;
   (c) ψ_i OR *falso* $\equiv \psi_i$

   Ad esempio, applicando queste regole, si ha:
   (NOT (Città = “Milano”))
   AND (Città = “Milano” OR Città = “Torino”`)
   AND (NOT (Città = “Torino”)) ≡ *falso*;

5. semplificazione della condizione eliminando predicati contraddittori (ad es., A > 20 AND A < 18 ≡ *falso*), eventualmente sfruttando le informazioni sui vincoli d'integrità;
6. eliminazione dell'operatore di negazione applicando le seguenti regole:
   (a) se l'operatore NOT precede una condizione semplice, si sostituisce la condizione con la sua complementare; ad esempio NOT (A > *Valore)* ≡ (A <= *Valore*);
   (b) se l'operatore NOT precede una condizione composta, si applicano le regole di De Morgan e si ripete il procedimento.

Nella fase di ottimizzazione algebrica, usando le proprietà algebriche degli operatori relazionali, si stabilisce l'ordine di applicazione degli operatori in modo da ridurre la dimensione dei risultati intermedi. Esempi di regole su cui si basa l'ottimizzazione sono:

1. Commutatività della restrizione e raggruppamento di condizioni

$$\sigma_X(\sigma_Y(R)) = \sigma_Y(\sigma_X(R)) = \sigma_{X \wedge Y}(R)$$

dove σ_X è l'operatore di restrizione con condizione sull'insieme di attributi X.

2. Raggruppamento di proiezioni

$$\pi_Z(\pi_Y(R)) = \pi_Z(R) \text{ , se } Z \subseteq Y$$

dove π_Z è l'operatore di proiezione sull'insieme di attributi Z

3. Commutatività della restrizione e della proiezione

$$\sigma_X(\pi_Y(R)) = \pi_Y(\sigma_X(R)) \text{ , se } X \subseteq Y$$

4. Commutatività della proiezione e della giunzione

$$\pi_Z(R \bowtie S) = \pi_{X \cap Z}(R) \bowtie \pi_{Y \cap Z}(S)$$

dove X sono gli attributi di R e Y sono gli attributi di S.

5. Commutatività della restrizione e della giunzione

$$\sigma_X(R \bowtie S) = \sigma_X(R) \bowtie S$$

se X sono gli attributi di R.

$$\sigma_{X \wedge Y}(R \bowtie S) = \sigma_X(R) \bowtie \sigma_Y(S)$$

se X sono gli attributi di R e Y sono gli attributi di S.

L'algoritmo di ristrutturazione procede secondo i seguenti passi:

1. Restrizioni con condizioni composte si separano in restrizioni con condizioni semplici usando la regola 1.
2. Si anticipa il più possibile l'esecuzione delle restrizioni usando le regole 3 e 5.
3. Si raggruppano restrizioni successive sulla stessa relazione usando la regola 1.
4. Si anticipa il più possibile l'esecuzione delle proiezioni usando le regole 2 e 4.
5. Si raggruppano proiezioni successive sulla stessa relazione e si eliminano proiezioni inutili usando la regola 2.

Ad esempio, applicando l'algoritmo all'espressione

$$\pi_{\text{Nome}}\left(\sigma_{\text{Provincia= "PI"} \wedge \text{ Materia= "DA"} \wedge \text{ Voto=30}}\left(\text{Studenti} \bowtie_{\text{Matricola=MatricolaCandidato}} \text{Esami}\right)\right)$$

si ottiene l'espressione

$$\pi_{\text{Nome}}\left(\left(\sigma_{\text{Provincia= "PI"}}(\text{Studenti})\right) \bowtie_{\text{Matricola=MatricolaCandidato}} \left(\sigma_{\text{Materia= "DA"} \wedge \text{Voto=30}}(\text{Esami})\right)\right)$$

Nella fase di ottimizzazione fisica, infine, usando le informazioni sulle strutture di memorizzazione e sulle statistiche disponibili, si stabilisce la strategia di esecuzione dell'interrogazione che ottimizzi il costo di esecuzione dell'interrogazione, e si genera il piano di accesso. Il procedimento viene illustrato considerando prima il caso di interrogazioni su un'unica relazione, cioè con soli predicati di restrizione, poi nel caso di interrogazioni su più relazioni, con predicati di restrizione e di giunzione.

Ottimizzazione fisica di interrogazioni su una relazione

Nelle interrogazioni su una relazione sono usate solo le operazioni di proiezione e restrizione dell'algebra relazionale. Gli algoritmi di sintesi differiscono nel modo in cui utilizzano gli indici disponibili e nell'ordine in cui le varie operazioni vengono eseguite.

Generalmente la condizione dell'interrogazione si considera nella forma ψ_1 AND ψ_2, dove ψ_1 è la sottocondizione coinvolgente attributi su cui è costruito un indice e ψ_2 è una sottocondizione la cui verifica necessita dell'accesso ai dati (predicati residui). Quindi, prima viene risolta la condizione ψ_1, utilizzando gli indici opportuni, poi le registrazioni vengono recuperate ed esaminate per verificare se soddisfano anche la condizione ψ_2.

In caso affermativo, esse faranno parte della risposta all'interrogazione, dopo essere state proiettate sugli attributi specificati. Ovviamente se non esistono indici sugli attributi coinvolti nella condizione, l'interrogazione viene risolta con una ***scansione sequenziale della relazione***.

Si presentano due tipi di algoritmi di accesso ai dati per la sintesi di un'interrogazione:

1. **algoritmo che usa al più un indice**: effettua l'accesso alle ennuple della relazione attraverso un solo indice, scelto tra quelli disponibili, oppure mediante la scansione sequenziale.
   Usato nel System R, e nei sistemi DB2 e SQL/DS da esso derivati, è preferito per non dover limitare l'accesso concorrente ai dati durante l'esecuzione delle operazioni sugli indici;

2. **algoritmo che usa più indici**: utilizza tutti gli indici utili, al fine di ridurre il numero delle ennuple da esaminare per trovare quelle che soddisfano l'interrogazione. Perciò esegue prima l'intersezione (o l'unione) dei riferimenti trovati mediante gli indici utili e, successivamente, accede alle ennuple per verificare i predicati rimanenti.

Questo metodo, adottato ad esempio da System 2000, ADABAS e ORACLE, tende quindi a minimizzare il numero degli accessi alla relazione, aumentando però gli accessi agli indici.

Algoritmo che usa al più un indice

Con l'algoritmo che usa al più un indice, si sceglie un indice tra quelli costruiti su attributi che compaiono in predicati di ricerca; successivamente si decide se è preferibile utilizzare l'indice o se procedere con la scansione sequenziale. I predicati di ricerca presi in considerazione devono però soddisfare l'ulteriore condizione di essere *fattori booleani*.

L'algoritmo consiste essenzialmente di due passi:

a) selezione di un predicato di ricerca;

b) controllo della condizione sui dati.

Selezione di un predicato di ricerca

Nel corso di questo passo occorre:

1. individuare i predicati di ricerca che sono anche fattori booleani;
2. valutare il costo di accesso con l'impiego dell'indice, per ogni predicato di ricerca;
3. scegliere il predicato di ricerca più conveniente, cioè quello di costo minimo.

Per eseguire questo passo, la condizione viene rappresentata con un albero, con i nodi terminali associati alle condizioni semplici e gli altri nodi associati agli operatori AND e OR.

I nodi associati ai predicati sono marcati con l'etichetta R, se si tratta di predicati di ricerca, altrimenti si marcano con l'etichetta N.

Si costruisce quindi l'insieme dei predicati di ricerca (LP) applicando il seguente algoritmo *ricorsivo*, a partire dalla radice dell'albero:

- se il nodo è un predicato di tipo R, allora LP = [Predicato],
- se il nodo è un predicato di tipo N, allora LP = [ ],
- se il nodo è AND, si applica l'algoritmo ricorsivamente ai sottoalberi sinistro e destro, producendo due insiemi che uniti danno l'insieme LP associato al nodo;
- se il nodo è OR, si procede come per il nodo AND, se entrambi gli insiemi del sottoalbero destro e sinistro non sono vuoti, altrimenti l'insieme LP associato al nodo è vuoto.

Alla fine dell'algoritmo, alla radice dell'albero è associato l'insieme dei predicati per i quali esiste un indice utile per la sintesi dell'interrogazione. A questo punto si eliminano dall'insieme LP tutti i predicati che hanno un antenato OR, producendo l'insieme LPA dei predicati di ricerca che sono anche fattori booleani. A questo punto si possono verificare tre casi:

- LPA è vuoto: non esistono indici da usare per limitare il numero di ennuple da visitare. Si marca allora la radice dell'albero con l'etichetta N (non risolto);
- LPA contiene un solo predicato: esiste allora un indice per fare la restrizione. Si marca la radice con S(semi-risolto), oppure con R (risolto) se tutta la condizione coincide con il predicato;
- LPA contiene più predicati: occorre allora scegliere il predicato più conveniente, stimando per ognuno di essi il costo di accesso $C_A = C_I + C_D$.
  Una volta selezionato un predicato di ricerca con costo minimo, la radice viene marcata con S.

Per stimare il costo di accesso associato a un predicato di ricerca, si stima il ***fattore di selettività*** della condizione, definito come il rapporto fra il numero di ennuple che soddisfano la condizione e il numero totale di ennuple presenti nella relazione.
Ad esempio, il fattore di selettività della condizione (sesso = "femminile") è circa 0,5 per l'insieme dei dati su una popolazione.
Nell'ipotesi di uniforme distribuzione dei valori di un attributo A_i sulle ennuple di una relazione, il fattore di selettività $f_s(\psi_{A_i})$ di una condizione su A_i è la probabilità che un'ennupla soddisfi la condizione e si stima come segue:

$$f_s(\psi_{A_i}) = \frac{card\left\{x \in dom(A_i) \mid x \ \ soddisfa \ \ la \ \ condizione \ \ su \ A_i\right\}}{N_{key}(A_i)}$$

In particolare si ha:

$$f_s(A_i = v) = \frac{1}{N_{key}(A_i)}$$

Se non si conosce $N_{key}(A_i)$ si può procedere come nel *System R* assumendo un fattore di selettività di 1/10 per ogni condizione ($A_i = v$).
Per attributi di tipo numerico, se si conoscono i valori massimo e minimo dell'attributo, si può porre:

$$f_s(A_i > v) = \frac{\max(A) - v}{\max(A) - \min(A)}$$

$$f_s(A_i < v) = \frac{v - \min(A)}{\max(A) - \min(A)}$$

$$f_s(A_i \text{ betwen } v_1 \text{ and } v_2) = \frac{v_2 - v_1}{\max(A) - \min(A)}$$

In assenza di informazioni, o quando l'attributo non è di tipo numerico, si assume un fattore di selettività di 1/3 per i primi due casi e di 1/4 nel terzo.

Questa scelta ha l'unico scopo di esprimere che, in assenza di informazioni, l'ottimizzatore assume una maggiore selettività (e quindi $f_s(\psi_{A_i})$ minore) per le condizioni di uguaglianza e selettività minori (e quindi $f_s(\psi_{A_i})$ maggiore) per le condizioni su intervalli di valori.

Per come è stato definito, è chiaro che tanto più il fattore di selettività è minore di uno, tanto più selettivo è il predicato cui si riferisce.

Dopo aver calcolato il fattore di selettività da assegnare a ciascun predicato di ricerca nell'insieme LPA, si calcola il costo di accesso all'indice e il costo di accesso ai dati.

Nell'ipotesi di uniforme distribuzione dei valori di A_i sul suo dominio, i valori di $f_s(\psi_{A_i})$ sono anche la frazione di foglie dell'indice che verranno visitate per trovare i riferimenti alle ennuple che soddisfano il predicato.

Quindi il costo d'accesso all'indice è dato da:

$$C_I = \lfloor f_s(\psi_{A_i}) \times N_{leaf}(A_i) \rfloor$$

Per quanto riguarda C_D, è necessario distinguere se l'indice è di ordinamento o non di ordinamento.

Nel primo caso, l'accesso attraverso l'indice di ordinamento si traduce in un accesso ordinato alle pagine dati, cosicché $f_s(\psi_{A_i})$ rappresenta anche la frazione di pagine dati cui si accede, quindi il costo di accesso ai dati è

$$C_D = \lfloor f_s(\psi_{A_i}) \times N_{pag}(R) \rfloor$$

Nel secondo caso, invece, assumendo la lista dei riferimenti ordinata, non si può verificare il caso che per un valore dell'attributo A_i una pagina dati venga visitata più volte, però si può verificare che una stessa pagina sia visitata più volte per valori dell'attributo compresi in un certo intervallo (più liste da visitare); per cui il costo si stima come

$$C_D = N.\,ListeDaVisitare \times N.\,PagineDaVisitare\ per\ ogni\ lista$$

$$C_D = \left\lfloor f_s(\psi_{A_i}) \times N_{key}(A_i) \times \Phi\left(N_t(A_i), N_{pag}(R)\right) \right\rfloor$$

dove Φ è la formula di Cardenas e $N_t(A_i)$ è il numero medio di ennuple per ogni valore dell'attributo A_i cioè $N_{reg}(R)/ N_{key}(A_i)$. Essendo $\Phi(k, m) < \min\{k, m\}$, il numero di pagine da visitare non supera mai N_{pag}.

Riassumendo, il costo globale di accesso è dato dalle seguenti espressioni:

$$C_A = \lfloor f_s(\psi_{A_i}) \times N_{leaf}(A_i) + f_s(\psi_{A_i}) \times N_{pag}(R) \rfloor$$

per l'indice di ordinamento e

$$C_A = \left[f_s(\psi_{Ai}) \times N_{leaf}(A_i) \right] + \left[f_s(\psi_{Ai}) \times N_{key}(A_i) \times \Phi\left(N_t(A_i), N_{pag}(R) \right) \right]$$

per un indice non di ordinamento.

Nel caso di una condizione contenente l'operatore **in**, conformemente alle ipotesi precedenti, si puo assegnare un fattore di selettività:

$$f_s\left(A_i \, in \left(v_1 ;.....; v_n \right) \right) = N \times f_s\left(A_i = v \right)$$

Mentre per il costo di accesso per indice e dati, non essendo possibile assumere contiguità alcuna tra due qualsiasi valori della condizione, il costo si stima come N volte quello per condizione (Ai=v):

$$C_A = N \times C_A\left(A_i = v \right)$$

NOTA 1

Se gli indici disponibili nel sistema non mantengono ordinato il gruppo di TID associato a ogni valore dell'attributo A_i, durante l'accesso alle ennuple di una relazione si può verificare il caso che per un valore dell'attributo una pagina dati venga visitata più volte, per cui la formula che si ottiene è la seguente:

$$C_D = \left\lfloor f_s(\psi_{Ai}) \times N_{reg}(R) \right\rfloor$$

Ciò fa diminuire fortemente l'utilità degli indici non di ordinamento in caso di predicati poco selettivi. Nel caso di indice costruito sull'attributo di ordinamento della relazione, il costo di accesso alle pagine dati è ancora $C_D = \left\lfloor f_s(\psi_{Ai}) \times N_{pag}(R) \right\rfloor$ poiché in questo caso i gruppi di TID sono ordinati come conseguenza dell'ordinamento della relazione.

NOTA 2

Se si usa un indice per accedere alle ennuple di una relazione, queste vengono reperite e restituite nell'ordine dei valori dell'attributo su cui è costruito l'indice.

Spesso però è richiesto che le ennuple del risultato siano in un ordine diverso; ad esempio, in SQL l'utente può specificare l'ordinamento che gli interessa con la clausola ***ORDER BY Attributo***.

Perciò se l'interrogazione contiene una di queste clausole, le ennuple reperite, prima di essere restituite all'utente, devono essere ordinate secondo l'ordine richiesto, se già non lo sono.

In questi casi è opportuno distinguere fra vie d'accesso (indici o scansione sequenziale) che producono l'ordinamento richiesto e vie d'accesso che non lo producono.

Per questo secondo tipo di vie d'accesso sarà necessario allora sommare al costo d'accesso calcolato con una delle formule viste, il **costo di ordinamento del risultato**.

Per poter dare una valutazione del costo di ordinamento è necessario però fare delle previsioni sul **numero** e la **dimensione** delle ennuple che fanno parte del risultato.

La dimensione delle ennuple può essere calcolata facendo la somma delle dimensioni degli attributi presenti nella *ListaDiAttributi* dell'interrogazione, su cui è appunto proiettato il risultato.

Invece il numero delle ennuple risultanti E_{reg} , duplicati inclusi, nell'ipotesi di uniforme distribuzione e di indipendenza dei valori di attributi differenti, può essere stimato con la seguente espressione:

$$E_{reg} = N_{reg}(R) \prod_{j=1}^{N_{FB}} f_s(FB_j)$$

dove N_{FB} è il numero dei fattori booleani di cui si può valutare la selettività, e $f_s(FB_j)$ è il fattore di selettività del j-esimo fattore booleano (in realtà la stima di E_{reg} diventa molto più complessa se si tiene conto del fatto che nel risultato vengono eliminate ennuple duplicate).

In questo calcolo bisogna considerare anche i predicati fattori booleani per i quali non esiste indice e quelli che non possono essere classificati come predicati di ricerca perché la condizione espressa non permette una ricerca selettiva attraverso un indice. Ad esempio predicati che contengono espressioni relazionali, congiunzioni e disgiunzioni su attributi diversi, per i quali è comunque possibile valutare il fattore di selettività:

$$f_s(A_1 = A_2) = 1 / \max\{N_{key}(A_1), N_{key}(A_2)\}$$

$$f_s(\psi A_1 \text{ AND } \psi A_2) = f_s(\psi A_1) \times f_s(\psi A_2)$$

$$f_s(\psi A_1 \text{ OR } \psi A_2) = f_s(\psi A_1) + f_s(\psi A_2) - f_s(\psi A_1) \times f_s(\psi A_2)$$

Infine, in base al numero e alla dimensione delle ennuple è possibile stimare il numero di pagine in cui sarà contenuto il risultato da ordinare e quindi il costo di ordinamento.

In generale un indice non di ordinamento sull'attributo A_i potrà essere preferito ad un indice di ordinamento, solo quando il predicato è molto selettivo, oppure quando è richiesto l'ordinamento del risultato secondo l'attributo A_i .

Una volta calcolati i costi di ciascun predicato e selezionato quello con costo minimo, siamo in grado di eseguire il passo successivo del metodo.

Controllo della condizione sui dati

Il procedimento di controllo della condizione sui dati dipende da come è marcata la radice dell'albero:

- se l'etichetta della radice è N, allora si procede con la scansione sequenziale della relazione e si estraggono le ennuple che soddisfano l'interrogazione.

- se l'etichetta della radice è S, si usa l'indice associato al predicato, ***solo se il suo costo è inferiore a quello della scansione sequenziale***, stimato pari al numero N_{pag} delle pagine della relazione; in caso contrario si procede con la scansione sequenziale;

- se l'etichetta è R, si procede come nel caso precedente; la differenza è che, se si usa l'indice, i riferimenti sono solo quelli delle ennuple che soddisfano l'interrogazione.

Riepilogo

Per quanto riguarda il costo del metodo, si hanno quindi due stime:

1. CA = N_{pag}, se l'etichetta della radice dell'albero è N,

2. se l'etichetta della radice dell'albero è S o R, supponendo di scartare la scansione sequenziale perché di costo maggiore, esistono due sottocasi:
   - se l'indice selezionato è di ordinamento:

$$C_A = \lfloor f_s(\psi_{A_i}) \times N_{leaf}(A_i) + f_s(\psi_{A_i}) \times N_{pag}(R) \rfloor$$

   - se l'indice selezionato è non di ordinamento:

$$C_A = \left[f_s(\psi_{A_i}) \times N_{leaf}(A_i) \right] + \left[f_s(\psi_{A_i}) \times N_{key}(A_i) \times \Phi\left(N_t(A_i), N_{pag}(R) \right) \right]$$

   oppure

$$C_A = \left[f_s(\psi_{A_i}) \times N_{leaf}(A_i) \right] + \left[f_s(\psi_{A_i}) \times N_{reg}(R) \right]$$

se le liste dei riferimenti nell'indice sono disordinate.

Esempio 1

Si consideri la relazione *Impiegati* (*Codice, Nome, Qualifica, Stipendio, Progetto, Città*), supponendo che:

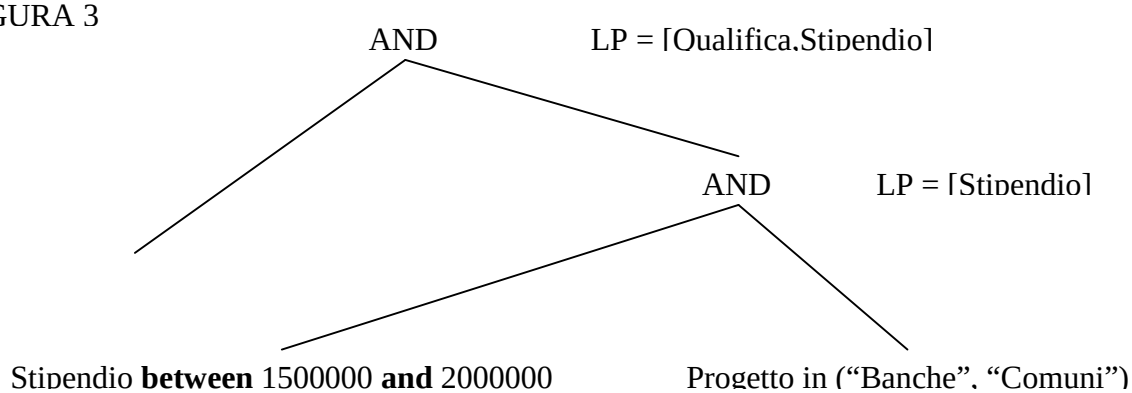
- sia ordinata su *Stipendio* con N_{reg} (*Impiegati*) =2000 e N_{pag} (*Impiegati*) =100;
- esista un indice di ordinamento su *Stipendio* con N_{key} (*Stipendio*) =200, N_{leaf} (*Stipendio*) =9, *min*(*Stipendio*) =1200000 e *max*(*Stipendio*) =2700000;

- esista un indice NON di ordinamento su *Qualifica* con N_{key} (*Qualifica*) =40 e N_{leaf} (*Qualifica*) =7;

select *Nome*, *Stipendio*
from *Impiegati*
where *Qualifica* = "Programmatore"
AND *Stipendio* **between** 1500000 **AND** 2000000
AND *Progetto* **in** ("Banche", "Comuni")

L'albero associato alla interrogazione è rappresentato nella figura 3:

FIGURA 3



Essendo LPA = [*Qualifica*, *Stipendio*], le alternative sono:

- scansione sequenziale;
- accesso con indice su Stipendio;
- accesso con indice su Qualifica.

I fattori di selettività dei predicati su *Qualifica* e *Stipendio* sono:

f_S(*Qualifica* = "Programmatore") = 1/40 = 0.025 e
f_S (*Stipendio* **between** 1500000 **and** 2000000) = 0.33

Essendo inoltre
N_t (*Qualifica*) = N_{reg} (*Impiegati*) / N_{key} (*Qualifica*) = 50

i costi delle tre alternative per l'accesso sono:

$C_A^{sequenziale}$ = 100

$C_A^{Qualifica}$ = [0.025 x 7] + (0.025 x 40 x Φ [50, 100] = 1 + 40 = 41

$C_A^{Stipendio}$ = [0.33 x 9] + [0.33 x 100] = 36

Viene preferito, quindi, ***l'accesso con l'indice su Stipendio***.

Questo esempio mostra come un predicato poco selettivo su un attributo di ordinamento possa dar luogo a un costo di scansione attraverso indice inferiore a quello prodotto usando un indice non di ordinamento con predicato molto più selettivo.

Ciò però non vale in generale, ad esempio se si avesse anche un indice non di ordinamento costruito su Progetto, per il quale sia N_{key} (*Progetto*) = 300 e N_{leaf} (*Progetto*) = 16, si avrebbe

f_S(*Progetto*) **in** ("Banche"; "Comuni") = 2/300 = 0.0066

N_t (*Progetto*) = 2000/300 = 7

da cui

$C_A^{Progetto}$ = [0.0066 x 16] + (0.0066 x 300 x Φ (7,100)] = 1 + 14 = 15

Esempio 2

Si consideri ancora la relazione *Impiegati* supponendo che:

- sia ordinata su *Città* con N_{reg} (*Impiegati*)= 10.000 e N_{pag} (*Impiegati*) = 1.000;
- esista un indice di ordinamento su *Città* con N_{key} (*Città*) = 20 e N_{leaf} (*Città*) = 30;
- esista un indice non di ordinamento su *Progetto* con N_{key} (*Progetto*) = 250 e N_{leaf} (*Progetto*) = 40;
- esista un indice non di ordinamento su *Codice* con N_{key} (*Codice*) = 10.000 e N_{leaf} (*Codice*) = 60;
- non esista un indice su *Stipendio* ma si conoscano i valori massimo e minimo: 1.000.000 e 5.000.000 di lire.

Si valuta il costo e il numero di ennuple trovate con la seguente interrogazione:

select *Nome, Stipendio*
from *Impiegati*
where *Città* = "Bologna"
AND *Stipendio* **between** 1000000 **AND** 1500000
AND *Progetto* = (**select** *Progetto*
from *Impiegati*
where *Codice* = Valore)

Viene prima eseguita l'interrogazione interna:

(**select** *Progetto*
from *Impiegati*
where *Codice* = Valore)

Essendo l'indice sul Codice per chiave primaria:

$C_A^{Codice} = 1+1=2$

Si considera ora l'interrogazione esterna:

select *Nome*, *Stipendio*

from *Impiegati*

where *Città* = “Bologna”

AND *Stipendio* **between** 1000000 **AND** 1500000

AND *Progetto* **=** *(valore trovato in precedenza)*

I fattori di selettività sono:

$f_S(Città = \text{"Bologna"}) = 1/ N_{key}(Città) = 1/20 = 0.05$

$f_S(Progetto = (\text{valore})) = 1/ N_{key}(Progetto) = 1/250 = 0.004$

Tutti i predicati sono fattori booleani ma sull’attributo *Stipendio* non esiste l’indice e quindi i costi delle possibili alternative sono:

- accesso con l’indice su *Città*:

$$C_A^{Città} = [f_S(Città = \text{"Bologna"}) \times N_{leaf}(Città)] + [f_S(Città = \text{"Bologna"}) \times N_{pag}(Impiegati)] = 52$$

- accesso con l’indice su *Progetto*:

$$N_t(Progetto) = N_{reg}(Impiegati) / N_{key}(Progetto) = 40$$

$$C_A^{Progetto} = [f_S(Progetto = (\text{valore})) \times N_{leaf}(Progetto)] + [f_S(Progetto = (\text{valore})) \times N_{key}(Progetto) \times \Phi(N_t(Progetto), N_{pag}(Impiegati)] = 1 + [1 \times \Phi(40, 1000)] = 40$$

Per la cardinalità del risultato bisogna valutare anche il fattore di selettività del predicato su *Stipendio*:

$$f_S(Stipendio \textbf{ between } 1000000 \textbf{ and } 1500000) = \frac{1500000 - 1000000}{5000000 - 1000000} = 0.125$$

Utilizzando la formula $E_{reg} = N_{reg}(R) \prod_{j=1}^{N_{FB}} f_s(FB_j)$=[10000 x 0.05 x 0.004 x 0.125] = 1

Algoritmo che usa più indici

Si usano tutti gli indici costruiti su attributi che compaiono in predicati di ricerca, costruendo per ciascuno una lista degli identificatori (TID) che soddisfano il predicato, quindi si effettua l'unione o l'intersezione di queste liste.
L'accesso alle ennuple viene effettuato attraverso riferimenti risultanti, scartando poi le ennuple che non soddisfano i predicati residui. L'algoritmo si articola in tre passi:
- selezione dei predicati di ricerca;
- combinazione delle liste di riferimento;
- controllo della condizione sui dati.

Selezione dei predicati di ricerca

In questo passo viene costruito l'insieme LP dei predicati di ricerca, che individuano gli indici utilizzabili per risolvere l'interrogazione, utilizzando lo stesso procedimento visto in precedenza.
Il quarto passo dell'algoritmo visto ("Selezione dei predicati di ricerca") ha importanza soprattutto in questo caso, dato che se si usa un solo indice i fattori booleani che hanno un OR come antenato sono eliminati.

Esempio 3
Si consideri la seguente interrogazione:
select *
from *Impiegati*
where *Nome* = "Rossi"
AND (*Qualifica* = "Programmatore" **OR** *Stipendio* = 1500000)

In questo caso l'utilizzazione di un indice su *Qualifica* non è di aiuto se non esiste un indice anche su *Stipendio*, perché tutte le ennuple fanno potenzialmente parte del risultato. In altre parole, nel caso di fattori booleani che contengono operatori OR, è necessario che tutti i predicati che intervengono siano risolubili con un indice.
Alla fine del passo di "Selezione dei predicati di ricerca" quindi, alla radice dell'albero della condizione è associata la sequenza dei predicati per i quali esiste un indice utile per la sintesi dell'interrogazione.

Combinazione delle liste di riferimento

Se la sequenza LP è vuota, si marca la radice dell'albero con N e si passa a eseguire il terzo passo. In caso contrario si visitano i nodi dell'albero e si marcano i nodi con i simboli R, N e S: se il nodo è risolto (R), ad esso viene associata la lista dei riferimenti di tutte le ennuple che soddisfano la condizione associata al nodo; se il nodo è semi-risolto (S), ad esso è associata la lista dei riferimenti di tutte le ennuple che potrebbero soddisfare la condizione associata al nodo, ma è necessario tuttavia accedere alla relazione per controllare quali dati effettivamente la soddisfino; se il nodo è non-risolto (N), ad esso non è associata alcuna lista di riferimenti.

La costruzione delle liste dei riferimenti è ottenuta con il seguente algoritmo ricorsivo a partire dalla radice dell'albero:

- se il nodo è un predicato di ricerca, cioè appartiene alla LP, si etichetta con R e si associa ad esso la lista dei riferimenti;
- se il nodo non è un predicato di ricerca, si etichetta con N;
- se il nodo è AND, la scelta dell'etichetta è fatta in base alle etichette dei suoi discendenti, utilizzando la seguente tabella

		etichetta discendente destro		
	AND	$R(l_d)$	$S(l_d)$	N
etichetta	$R(l_S)$	$R(l_S \cap l_d)$	$S(l_S \cap l_d)$	$S(l_S)$
discendente	$S(l_S)$	$S(l_S \cap l_d)$	$S(l_S \cap l_d)$	$S(l_S)$
sinistro	N	$S(l_d)$	$S(l_d)$	N

- se il nodo è OR, la scelta dell'etichetta è fatta in base alle etichette dei suoi discendenti, utilizzando la seguente tabella

		etichetta discendente destro		
	OR	$R(l_d)$	$S(l_d)$	N
etichetta	$R(l_S)$	$R(l_S \cup l_d)$	$S(l_S \cup l_d)$	N
discendente	$S(l_S)$	$S(l_S \cup l_d)$	$S(l_S \cup l_d)$	N
sinistro	N	N	N	N

Alla conclusione di questo passo si troverà associato alla radice dell'albero una delle seguenti informazioni:

- N, condizione non risolta;
- S[*ListaDeíRiferimenti*], condizione semirisolta;
- R[*ListaDeiRiferimenti*], condizione risolta.

Esempio 3

Si consideri la relazione *Impiegati* precedentemente definita e l'interrogazione

select *

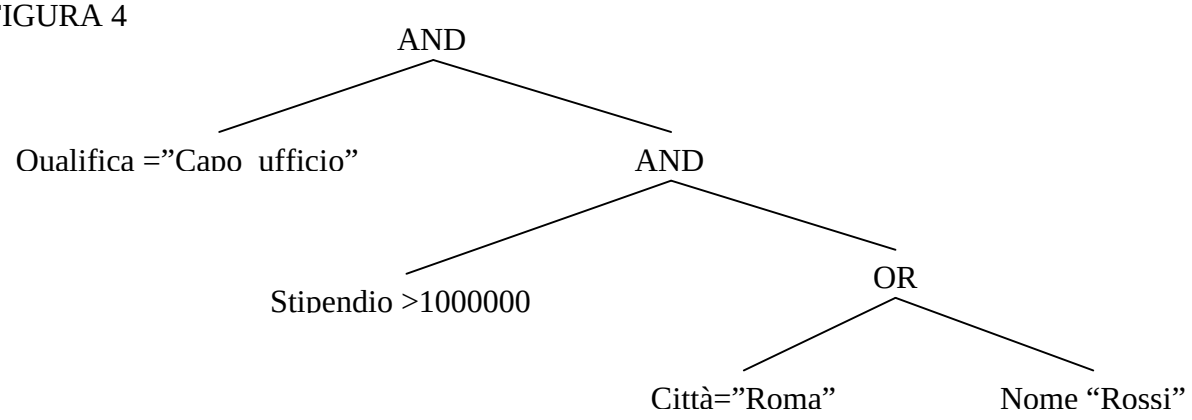
from *Impiegati*

where *Qualifica* = "Capo ufficio" **AND** *Stipendio* > 1.000.000

AND (*Città* = "Roma" **OR** *Nome* ≠ "Rossi")

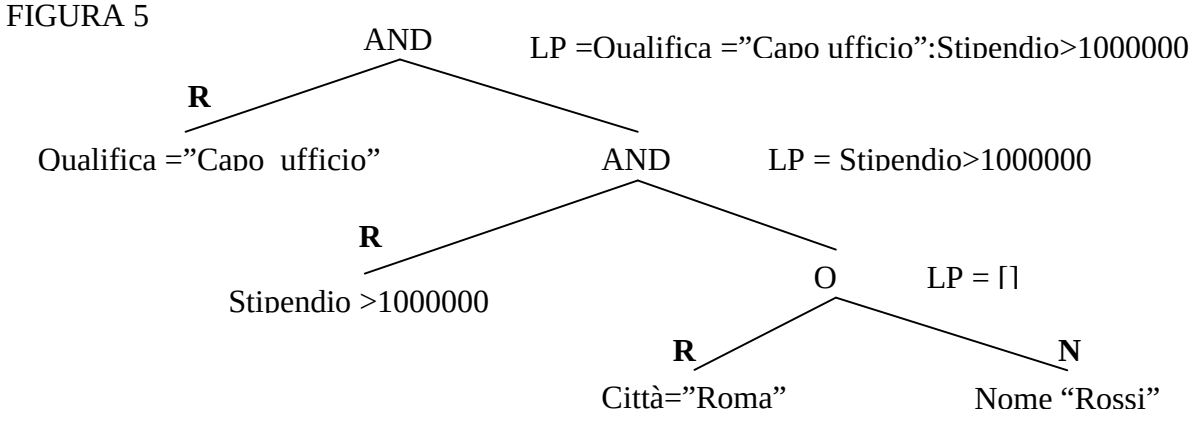
L'albero della condizione è rappresentato nella figura 4

FIGURA 4



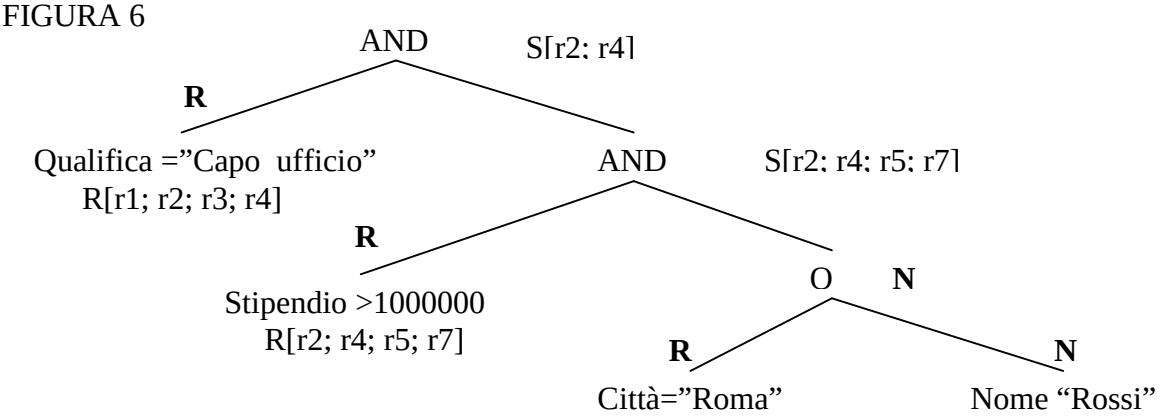
Dopo il primo passo della selezione dei predicati di ricerca, supponendo che siano definiti indici secondari su *Qualifica*, *Stipendio* e *Città* e di ordinamento su *Nome*, si ottiene l'albero rappresentato nella figura 5, in cui il nodo terminale *Nome* ≠ "Rossi" non è un predicato di ricerca e di conseguenza anche città = "Roma" non compare nella sequenza LP dei predicati perché è legato da un OR a un predicato non risolubile

FIGURA 5



Infine dopo il passo di combinazione e ordinamento delle liste dei riferimenti, si ottiene l'albero rappresentato nella figura 6

FIGURA 6



Controllo della condizione sui dati

In base all'etichetta associata alla radice dell'albero, si decide come accedere alla relazione per verificare quali ennuple soddisfino l'interrogazione. Esistono tre alternative:

- se l'etichetta è N, è necessario scandire la relazione per trovare le ennuple che soddisfano l'interrogazione ;
- se l'etichetta è S, è necessario accedere alle ennuple della lista dei riferimenti per verificare quali di esse soddisfano l'interrogazione ;
- se l'etichetta è R, la lista dei riferimenti contiene i riferimenti di tutte e sole le ennuple che soddisfano l'interrogazione. L'accesso ai dati è richiesto per produrre il risultato e fare le eventuali proiezioni sugli attributi specificati.

Per quanto riguarda il costo dell'algoritmo, si hanno quindi due stime:

1. se l'etichetta della radice è N, poiché è necessario eseguire la scansione della relazione, il costo è

$$C_A = N_{pag}(R)$$

2. se l'etichetta della radice è S o R, supponendo che le operazioni sulle liste avvengano in memoria temporanea, il costo globale di accesso agli indici è

$$C_A = \sum_{k=1}^{N_{pred}} C_I(A_K)$$

dove $C_I(A_K)$ è il costo d'accesso attraverso l'indice per risolvere un predicato di ricerca, calcolato con la formula $C_I(A_K) = \lfloor f_S(A_K) \times N_{leaf}(A_K) \rfloor$, precedentemente ricavata. La sommatoria è estesa a tutti i predicati di ricerca.

Per poter calcolare il numero di accessi alle pagine dei dati è necessario avere una stima del numero E_{reg} di riferimenti, e quindi di ennuple a cui bisogna accedere, usando la formula sopra vista:

$$E_{reg} = N_{reg}(R) \prod_{j=1}^{N_{FB}} f_s(FB_j)$$

Supponendo di adottare la tecnica di ordinare la lista dei riferimenti risultante, il numero di accessi alle pagine dati è $C_D = \Phi(E_{reg}, N_{pag}(R))$.

Se la lista non è ordinata, il numero di accessi alle pagine è dato dal numero di ennuple selezionate:

$$C_D = E_{reg}$$

Osservando la formula per il calcolo di E_{reg}, si vede che E_{reg} diminuisce al crescere del numero degli indici usati.

D'altra parte C_I aumenta per ogni indice introdotto, ***quindi si può arrivare alla situazione in cui un indice produca un vantaggio minore del costo aggiuntivo dato dal suo uso***.

Questo può accadere per esempio quando l'indice viene usato con predicati poco selettivi.

Per ovviare a questo inconveniente si può adottare l'accorgimento proposto:

1. per ogni indice si calcolano i costi C_I e C_D come per il metodo che usa al più un indice;

2. si ordinano gli indici sulla base del costo globale ($C_I + C_D$) in maniera crescente;

3. si considera un indice I_j per volta confrontando il costo aggiuntivo con il beneficio che se ne ricava $\Phi(E'_{reg}, N_{pag}(R)) - \Phi(E''_{reg}, N_{pag}(R))$, dove E'_{reg} ed E''_{reg} sono i risultati ottenuti in assenza e in presenza dell'indice I_j;

4. se il costo aggiuntivo è maggiore del beneficio si scarta l'indice altrimenti si prosegue con il passo 3.

Anche se l'algoritmo che usa più indici può dare luogo a un minor numero di accessi ai dati rispetto al metodo che usa un solo indice, quest'ultimo viene preferito (ad es. nei prodotti IBM) per due motivi:

- l'elaborazione di liste di TID e il loro ordinamento può dar luogo, per relazioni di grandi dimensioni, a un considerevole costo aggiuntivo se le liste non possono essere elaborate in memoria temporanea;
- nel caso di un sistema in multiutenza, l'uso di più indici può richiedere algoritmi di blocco dei dati molto complessi, oppure il blocco di tutti gli indici quando l'utente richiede un accesso di tipo esclusivo.

Ottimizzazione fisica di interrogazioni su più relazioni

Si esamina il problema dell'ottimizzazione fisica di interrogazioni che prevedono la giunzione di più relazioni. L'operazione di giunzione è una delle operazioni più costose, sia come tempo di CPU che come numero di accessi alla memoria permanente. Sono stati studiati molti metodi per l'esecuzione delle giunzioni e spesso i sistemi ne adottano più di uno fra i quali l'ottimizzatore sceglie quello più conveniente in ogni circostanza.

Si esamina il caso di interrogazioni con un predicato di giunzione su due relazioni, per il quale sono stati proposti molti metodi di risoluzione, ma solo due di essi, con qualche variante, hanno trovato applicazione nei sistemi commerciali perché sono risultati quelli che hanno in generale prestazioni migliori: il ***nested loop*** e il ***sort-merge***.

Prima di analizzarli, si definisce il significato di alcune abbreviazioni che si useranno frequentemente. **Un indice di giunzione è un indice costruito sull'attributo coinvolto nel predicato di giunzione (attributo di giunzione)**. **Analogamente, un indice di restrizione è un indice costruito su un attributo coinvolto in un predicato di restrizione (attributo di restrizione)**. Con riferimento all'ordine in cui vengono considerate le relazioni nell'operazione di giunzione, **si definisce esterna la relazione che verrà considerata per prima**, interna l'altra.

Metodo nested loop

Con il metodo nested loop si accede alla relazione esterna R, usando indici o con una scansione sequenziale, e per ogni ennupla che soddisfa la ψ_R si accede alla relazione interna S per cercare le ennuple che soddisfano la ψ_S e quella di giunzione $R.A_i = S.A_j$. Le coppie così ottenute vengono proiettate sugli attributi richiesti e il risultato è presentato nell'ordine della relazione esterna.

La relazione R viene quindi scandita una sola volta, mentre S tante volte quante sono le ennuple di R che soddisfano la ψ_R. Il costo dell'operazione è:

$$C_{NL} = C_A(R) + E_{reg}(R) \times C_A(S)$$

dove $E_{reg}(R)$ è il numero di ennuple che soddisfano la ψ_R. Tutte le quantità si stimano con le formule viste in precedenza.

Il costo globale in generale varia scambiando la relazione esterna con quella interna, e questa possibilità viene presa in considerazione dall'ottimizzatore.

Si supponga ad esempio di avere un'operazione di giunzione senza predicati di restrizione e di procedere secondo lo schema previsto dal metodo nested loop:

for *r* **in** *R* **do**

    for *s* **in** *S* **with** $r.A_i = s.A_j$ **do**

        unisci le ennuple *r* e *s*

Il numero di pagine visitate considerando R esterna è $N_{pag}(R) + N_{reg}(R) \times N_{pag}(S)$
Considerando invece S come esterna, il numero di pagine visitate è $N_{pag}(S) + N_{reg}(S) \times N_{pag}(R)$.
Le due quantità sono in generale diverse.
Nell'ipotesi che siano disponibili indici sugli attributi di giunzione, uno solo di questi è utilizzabile per agevolare l'operazione di giunzione, quello sulla relazione interna.
Infatti un predicato di giunzione del tipo $R.A_i = S.A_j$ può essere considerato argomento di ricerca solo per la relazione interna S, in quanto solo per essa esistono disponibili i valori da sostituire nell'uguaglianza, ottenuti accedendo di volta in volta alle ennuple della esterna R.
Per S quindi il predicato sull'attributo di giunzione deve essere valutato dall'ottimizzatore insieme agli altri predicati locali di restrizione. Per R, invece, l'unico caso in cui è opportuno prendere in considerazione anche l'uso dell'indice su $R.A_i$ è quando $R.A_i$ compare in una clausola di ORDER BY o GROUP BY.
L'algoritmo per la risoluzione dell'interrogazione prevede allora che
(a) si scelga l'indice di minor costo per la ricerca delle ennuple della relazione esterna che soddisfano i predicati locali di restrizione e
(b) si scelga l'indice di minor costo sulla relazione interna per trovare le ennuple che soddisfano sia i predicati locali che quello di giunzione.
Una variante del metodo, denominata nested block, procede secondo lo schema:
for pagina *r* **in** *R* **do**

    for pagina *s* **in** *S* **do**

        unisci le ennuple in *r* e *s* che soddisfano il predicato di giunzione

In assenza di indici, il numero di pagine visitate considerando R esterna è

$$N_{pag}(R) + N_{pag}(R) \times N_{pag}(S)$$

Considerando invece S come esterna, il numero di pagine visitate è

$$N_{pag}(S) + N_{pag}(S) \times N_{pag}(R)$$

Le due quantità differiscono nel primo termine e il costo dell'operazione sarà pertanto minore quando si usa come esterna la relazione che occupa meno pagine.
Questo metodo in assenza di indici è senz'altro migliore del *nested loop*, essendo $N_{pag}(R) < N_{reg}(R)$, ha però il difetto di perdere un eventuale ordinamento del risultato basato sulla scansione ordinata della prima relazione.

Se il buffer ha capacità di $B \geq 2$ pagine, (B - 1) pagine si assegnano alla relazione esterna R e una a quella interna S. Ad ogni ciclo, per ogni (B - 1) pagine di R si leggono tutte le pagine di S, che viene quindi visitata $[N_{pag}(R) /(B-1)]$ volte.
Il costo del metodo diventa

$$N_{pag}(R) + [N_{pag}(R) /(B-1)]x\ N_{pag}(S)$$

Metodo nested loop con creazione di relazioni temporanee

Si procede in due passi:
a) le relazioni R ed S vengono ristrette, usando i predicati locali, e proiettate sugli attributi richiesti e sugli attributi di giunzione; i risultati vengono memorizzati in due relazioni temporanee R' e S';
b) si esegue la giunzione di R' ed S'.
La creazione di R' ed S' può essere effettuata scegliendo gli indici migliori utilizzabili con i predicati locali (**si noti che questo metodo non sfrutta l'eventuale presenza di indici sugli attributi di giunzione**). Se le due relazioni temporanee sono grandi, le pagine che le contengono devono essere portate in memoria permanente per poi essere riprese per effettuare la giunzione. In generale il costo totale dell'operazione è:

$$C_{NLT} = C_A(R) + C_A(S) + [N_{pag}(R') + N_{pag}(S') + [N_{pag}(R') + E_{reg}(R') \text{ x } N_{pag}(S')]]$$

La componente del costo tra parentesi quadre manca quando le due relazioni temporanee sono tenute nel buffer.
La valutazione di $E_{reg}(R')$ e $E_{reg}(S')$, e quindi di $N_{pag}(R')$ e $N_{pag}(S')$, non è sempre agevole poiché, mentre è semplice valutare il numero di ennuple selezionate da una restrizione e la dimensione di una ennupla proiettata su alcuni attributi, non è sempre facile la previsione del numero di ennuple risultanti da una proiezione in quanto essa elimina le ennuple uguali.
La componente del costo riguardante la giunzione aumenta con $N_{pag}(R')$ e $N_{pag}(S')$ perché le due relazioni temporanee sono senza indici, pertanto questo metodo è conveniente se i predicati locali e le proiezioni sono molto selettivi.
L'uso di relazioni temporanee può essere combinato con una tecnica di trasformazione della chiave. Le ennuple di R che soddisfano la ψ_R si memorizzano in una relazione temporanea con una funzione hash H applicata all'attributo di giunzione. Successivamente, a ogni ennupla di S che soddisfa la ψ_S si applica la funzione hash al valore dell'attributo di giunzione per recuperare l'ennupla della relazione temporanea da unire per produrre il risultato.

Il costo è

$$C_A(R) + E_{reg}(R) + C_A(S) + E_{reg}(S)$$

Metodo sort-merge

Con il metodo *sort-merge* si assume che le relazioni siano ordinate sugli attributi di giunzione, se non lo sono è necessario procedere preventivamente all'ordinamento creando due relazioni temporanee.

L'algoritmo procede scandendo entrambe le relazioni per valori crescenti dell'attributo di giunzione. Ogni volta che viene individuata una coppia di ennuple che verificano la ψ_R, la ψ_S e il predicato di giunzione, la coppia viene proiettata sugli attributi richiesti ed entra a far parte del risultato. L'ordinamento delle relazioni consente di scandire entrambe le relazioni una sola volta. Supponendo che le ennuple della relazione interna con lo stesso valore dell'attributo di giunzione siano in un'unica pagina, il costo dell'operazione è:

$$C_{SM} = ord(R) + ord(S) + N_{pag}(R) + N_{pag}(S)$$

con ord(R) e ord(S) l'eventuale costo di ordinamento e creazione delle relazioni temporanee.
Se le relazioni non sono ordinate sugli attributi di giunzione, ma sono definiti su di essi degli indici, il metodo potrebbe essere applicato scandendo le relazioni attraverso questi indici.
Questa opportunità non viene quasi mai usata perché la scansione di una relazione attraverso un indice definito su un attributo non di ordinamento può avere costi elevati.

Esempio 4

Si considerino le relazioni *Impiegati* (*Codice, Nome, Qualifica, Città, Dipartimento*) e *Dipartimenti*(*Sigla, Denominazione, Località, Direttore*) e l'interrogazione:

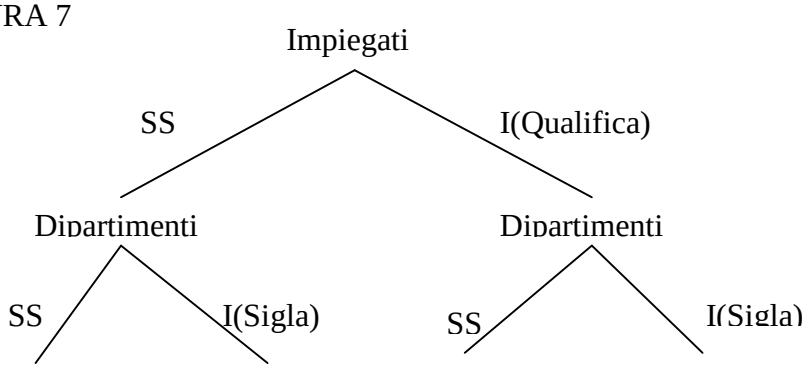
select *Nome, Direttore*
from *Impiegati, Dipartimenti*
where *Sigla = Dipartimento*
AND *Qualifica* = "Analista" **AND** *Località* = "Roma"

Si supponga che esistano indici su *Dipartimento, Qualifica* e *Sigla*, e che si usi al più un indice per restringere una relazione.
Per decidere come generare il risultato, per ciascun metodo di giunzione disponibile e per ciascun ordine di accesso alle relazioni (prima *Impiegati* o *Dipartimenti*), si procede nel modo seguente:

- si calcolano i costi delle vie di accesso alle singole relazioni, tenendo presente che i predicati da isolare sono di due tipi: quelli locali e di giunzione, questi ultimi utilizzabili solo per la relazione interna. Infatti, un predicato di giunzione del tipo $R.A_i = S.A_j$, con R relazione esterna, è utilizzabile solo per S poiché, come è già stato mostrato in precedenza, soltanto dopo aver effettuato l'accesso a un'ennupla di R è noto il valore di $R.A_i$ e il predicato di giunzione diventa del tipo $v = S.A_j$;
- per ogni relazione, viene selezionata la via d'accesso di minor costo, quella sull'attributo di giunzione e quelle che riguardano le clausole di ordinamento;
- vengono calcolati i costi globali, considerando le due possibilità per la relazione esterna;
- si sceglie la soluzione di costo minimo.

Si presenta un possibile sviluppo dell'esempio limitandosi al metodo *nested loop*, considerando come relazione esterna *Impiegati*; le alternative sono riportate nella Figura 7:

FIGURA 7



SS = scansione sequenziale
I(A) = indice su attributo A

Bisogna scegliere fra accesso con indice su *Qualifica* e la scansione sequenziale.
Supponendo che sia più conveniente l'indice su *Qualifica*, si esclude il sottoalbero con radice la scansione sequenziale. Supponendo di escludere la scansione sequenziale anche per *Dipartimenti*, il costo globale risulta:

$$C = C_A^{Qualifica} + E_{reg}(\text{Impiegati}) \times C_A^{Sigla}$$

Esempio 7

Si considerino le relazioni:

- *Prodotti*(*PN*, *Pnome*, *Ptipo*, *PJ*, *Altro*), con chiave (*PN*, *PJ*), N_{reg}(*Prodotti*) = 2.000 e N_{pag}(*Prodotti*) = 400;
- *Progetti*(*PJ*, *Nome*, *Dno*, *Altro*), con chiave PJ, N_{reg}(*Progetti*) = 200 e N_{pag}(*Progetti*) = 100;

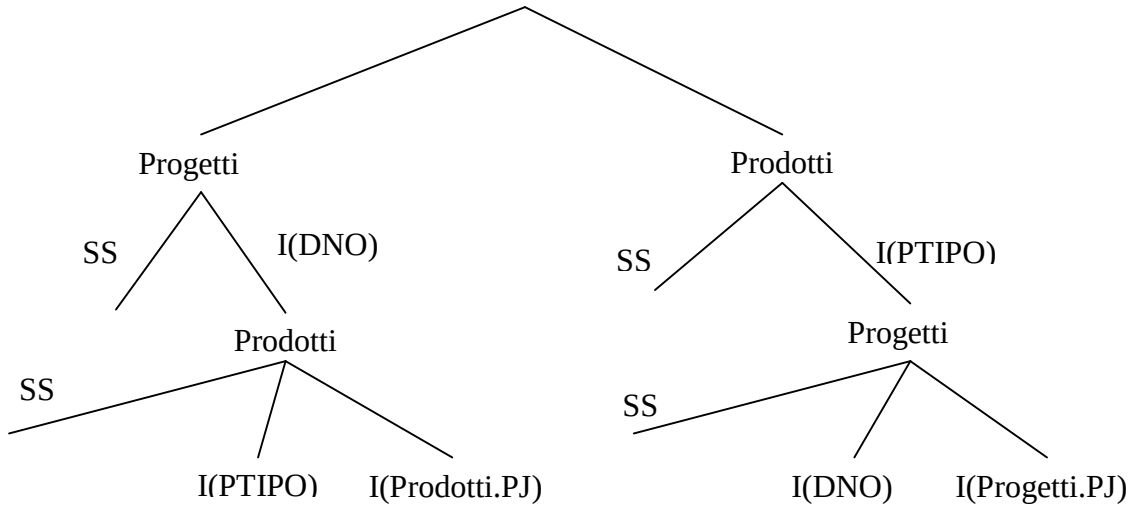
e l'interrogazione:

select *PJ, Nome*
from *Progetti, Prodotti*
where *Progetti.PJ = Prodotti.PJ*
AND *Dno* = 13 **AND** *Ptipo* = "AA"

Si supponga che esistano i seguenti indici non di ordinamento con gruppi di TID ordinati:
- su *Dno* con N_{key}(*Dno*) = 20 ed N_{leaf}(*Dno*) = 3;
- su *Ptipo* con N_{key}(*Ptipo*) = 100 ed N_{leaf}(*Ptipo)* = 20;
- su *PJ* in *Prodotti* (indicato con *Prodotti.PJ*) con N_{key}(*Prodotti.PJ*) = 200 ed N_{leaf}(*Prodotti.PJ*)=25;
- sulla chiave *PJ* in *Progetti* (*Progetti.PJ*) con N_{key}(*Progetti.PJ*) = 200 ed N_{leaf}(*Progetti.PJ*) = 5.

Le alternative usando il metodo *nested loop* sono mostrate nella figura 8:

FIGURA 8



SS = scansione sequenziale
I(A) = indice su attributo A

1. Relazione esterna *Progetti.*
   Il costo di accesso con l'indice su *Dno* è:

   f_S (*Dno*) = 13)) = 1/20, N_t(*Dno*) = 10
   C_A^{Dno} = 3/20 + Φ (10,100) = 1 + 10 = 11

che è inferiore al costo di scansione sequenziale N_{pag}(*Progetti*) = 100.
Il numero di ennuple risultanti dalla selezione è E_{reg}(*Progetti*) = 10.

Il costo per l'accesso a *Prodotti* con l'indice su *Ptipo* è:

f_S(*Ptipo* = "AA") = 1/20, N_t(*Ptipo*) = 20
C_A^{Ptipo} = 20/100 + Φ (20,400) = 1 + 20 = 21

e per l'indice su *Prodotti.PJ*:

f_S(*Prodotti.PJ* = *valore*) = 1/200, N_t(*Prodotti.PJ*) = 10
$C_A^{Prodotti.PJ}$ = 25/200 + Φ (10, 400) = 1 + 10 = 11

Entrambi i costi sono inferiori a quello della scansione sequenziale N_{pag}(*Prodotti*) = 400.
Il costo totale è quindi:

$$C_A^{Dno} + E_{reg}(Progetti) \times C_A^{Prodotti.PJ} = 11 + 10 \times 11 = 121$$

2. Relazione esterna *Prodotti*.
   Il costo di accesso con l'indice su *Ptipo* è:
   C_A^{Ptipo} = 21, come già calcolato, ed E_{reg}(*Prodotti*) = 20.

   Il costo di accesso a *Progetti* con l'indice su *Dno* è:
   C_A^{Dno} = 11, come già calcolato, mentre quello con l'indice su *Progetti.PJ* è:

   $C_A^{Progetti.PJ}$ = 1 + 1 = 2

   Il costo totale è quindi:
   $C_A^{Dno} + E_{reg}(Prodotti) \times C_A^{Progetti.PJ}$ = 21 + 20 x 2 = 61 che è inferiore a quello del caso Precedente.

Esempio 7

select *

from *R1,R2*

where *R1.C = R2.C*

AND *R1.A* = 574 **AND** *R2.B* < 800

Supponendo $N_{reg}(R1) = 10000$, $N_{reg}(R2) = 500$, $N_{pag}(R1) = 1000$, $N_{pag}(R2) = 100$ e che esistano i seguenti indici:

- indice di ordinamento su *R2.C* con $N_{key}(R2.C) = 500$ e $N_{leaf}(R2.C) = 10$;
- indice di non ordinamento su *R1.C* con liste di riferimenti disordinate, $N_{key}(R1.C) = 500$ e $N_{leaf}(R1.C) = 1000$;
- indice di ordinamento su *R1.A* con $N_{key}(R1.A) = 1000$ e $N_{leaf}(R1.A) = 100$.

Procedendo come nell'esempio precedente, si trova che con il metodo *nested loop* accedendo prima a *R1*con l'indice *R1.A* e poi a *R2* con l'indice *R2.C* si ha un costo di 22.
Visitando invece prima *R2* e, per ogni ennupla che soddisfa la condizione *R2.B* < 800, si accede alle ennuple di *R1* che soddisfano la condizione di giunzione usando l'indice *R1.C* e si controlla la condizione *R1.A* = 574, si ha un costo

$$\begin{aligned} C_A &= C_A(R2) + E_{reg}(R2) \times C_A(R1) \\ &= N_{pag}(R2) + [f_S(R2.B < 800) \times N_{reg}(R2)] \times \\ &\quad [[f_S(R1.C = valore) \times N_{leaf}(R1.C)] + \\ &\quad [f_S(R1.C = valore) \times N_{reg}(R1)]] \\ &= 100 + [500/3] \times [[100/500] + [10000/500]] = 3607 \end{aligned}$$

Un'altra possibilità è di applicare il metodo *sort-merge* dopo aver ordinato *Rl* secondo l'attributo *R1.C*, con un costo pari a $2 \times N_{pag}(R1)(1 + [\log_Z(N_{pag}(R1)])$, con z = 2:

$$C_A^{SM} = N_{pag}(R2) + ord(R1) + N_{pag}(R1) = 100 + 22000 + 1000 = 23100$$

Il metodo più conveniente è quindi il *nested loop* con Rl interna.

Nel caso di giunzione fra più di due relazioni, il numero delle alternative da valutare aumenta sensibilmente. Ad esempio, per eseguire la giunzione di tre relazioni *R(A, B)*, *S(B, C)* e *T(C, D)*, **occorre prendere in considerazione tutte le permutazioni dell'ordine di giunzione eliminando quelle che comportano un prodotto di relazioni**.
Quindi due alternative valide sono (R ⋈ S) ⋈ T e R ⋈ (S ⋈ T), ma va scartato l'ordine (R ⋈ T) ⋈ S perché comporterebbe un prodotto cartesiano.
Per ciò che riguarda la giunzione che viene eseguita per prima i sistemi DB2 e SQL/DS nel caso del metodo *sort-merge* prevedono la costruzione di una relazione temporanea, mentre nel caso *nested*

loop ogni ennupla della prima giunzione viene subito confrontata con le ennuple della terza relazione senza creazione di relazioni temporanee.

Query Optimizer

Il modulo di gestione di database **Microsoft Jet** contiene numerosi componenti, ma quello più importante per le query, oltre che il più complesso, è **Query Optimizer**. Si tratta di un'utilità di ottimizzazione ***basata sul costo*** che assegna pertanto a ciascuna attività di query un costo in termini di tempo scegliendo alla fine l'elenco delle attività *meno dispendioso* da eseguire per generare l'insieme di risultati richiesti. Più tempo è richiesto per l'esecuzione di un'attività, più costosa o dispendiosa è considerata tale attività.

Per decidere quale strategia di query utilizzare, Query Optimizer utilizza delle statistiche basate sul numero di record e di pagine di dati presenti in una tabella, sull'ubicazione della tabella, sulla presenza o meno di indici, sulla loro univocità e così via. Sulla base di queste statistiche viene quindi scelta la strategia di query interna migliore per svolgere una particolare query.

Le statistiche vengono aggiornate ogni volta che si compila una query. Una query viene contrassegnata per la compilazione ogni volta che si salvano delle modifiche alla query o alle tabelle sottostanti e quando il database viene compattato. Se una query viene contrassegnata, la compilazione e l'aggiornamento delle statistiche verranno eseguiti in occasione della successiva esecuzione della query. La compilazione di una query richiede solitamente da uno a quattro secondi.

Se viene aggiunto un numero significativo di record al database, si consiglia di aprire e salvare le query per ricompilarle. Ad esempio, se si progetta e si sottopone a verifica una query utilizzando un piccolo insieme di dati di esempio, si consiglia di ricompilare la query dopo aver aggiunto ulteriori record al database. In questo modo si garantiranno prestazioni ottimali della query quando l'applicazione è in uso.

NOTA: non è possibile visualizzare gli schemi di ottimizzazione del modulo di gestione di database Jet né specificare le modalità di ottimizzazione di una query. Ciononostante è possibile utilizzare Database Documenter per scoprire se sono presenti indici nelle tabelle di database e se tali indici sono univoci.

APPENDICE

Appendice A.1

TABELLE DI SERVIZIO

Paesi:

- **Cod_Paese** (Testo)
- Des_Paese (Testo)

Giorni : la tabella ha chiave multipla. Contiene il calendario di ciascun paese con relative festività secondo l'anno solare.

- **Cod_Paese** (Testo)
- **Giorno** (Data/ora)
- Status (Testo)

Descrizioni_Feste :

- **Cod_Festa** (Testo)
- Des_Festa (Testo)

Divise : la tabella contiene i codici delle divise di conto.

- **Cod_Divisa** (Testo)
- Des_Divisa (Testo)
- Base_Currency (Sì/No)

Mercati : la tabella contiene i codici dei principali mercati di trattazione degli strumenti finanziari (Nasdaq, Nyse, ecc.).

- **Cod_Mercato** (Testo)
- Cod_Paese (Testo)
- Des_Mercato(Testo)

Operatori :

- **Cod_Operatore** (Testo)
- Operatore (Testo)
- Des_Operatore (Testo)

Entita' : la tabella contiene l'elenco completo di tutti i soggetti finanziari coinvolti nella trattazione di ciascuno strumento finanziario visti , oltre alle agenzie di rating atte alla valutazione del grado di solvibilità e affidabilità dei soggetti stessi.

- **Cod_Entita** (Contatore)
- Cod_Entita_Utente (Testo)
- Cod_ABI_AG (Testo)
- Des_Entita (Testo)
- Cod_tipo_entita (Testo)
- Cod_Settore (Testo)
- Cod_Paese (Testo)
- Cod_Agenzia_Rating (Numerico)
- Cod_Rating (Testo)

Tipi_Entita' : contiene le tipologie di entità previste

- **Cod_Tipo_Entita** (Testo)
- Des_Tipo_Entita (Testo)

Rating : la tabella è a chiave multipla.

- **Cod_Entita** (Numerico)
- **Cod_Rating** (Testo)
- Basis_Point (Numerico)
- Des_Rating (Testo)
- Des_Agenzia_Rating (Testo)

Analisi_Mod : elenco delle modellizzazioni consentite

- **Cod_Mod_Analisi** (Contatore)
- Mod_Analisi (Testo)
- Des_Mod_Analisi

Tipo_Strumento : ogni singolo record della tabella (quindi ogni singolo tipo di strumento), è univocamente identificato dalla combinazione di quattro campi, per un totale di 74 tipi di strumenti finanziari possibili.

- Cod_Tipo_Strumento (Contatore)
- Des_Tipo_strumento (Testo)
- **Classe_Strumento** (Numerico)

- **Quotato/Non_Quotato** (Testo)
- **Nozionale/Quantità** (Testo)
- **Euro/Non_Euro** (Testo)

Classe_Strumento :

- **Cod_Classe** (Numerico)
- Cod_Classe (Testo)

Accrual_Convention : la tabella ha chiave doppia; riassume le convenzioni di calcolo su date che si possono utilizzare.

- **ID_AccConv** (Contatore)
- **AccrualConvention** (Testo)
- Des_Acc_Conv (Testo)

Info_Provider :

- **Id_Provider**
- Des_Info_Provider

Tipologie_Portafogli :

- **Cod_Tipologia_Portafoglio**
- Des_Tipologia_Portafolgio

Generi_Portafogli : il portafoglio di proprietà può essere di genere immobilizzato, non immobilizzato o di negoziazione; mentre quello gestito può essere di genere fondo pensione, fondo comune, GPM, assicurativo o altro.

- **Cod_Genere** (contatore)
- Cod_Tipologia (Numerico)
- Des_Genere

Tipo_Opzione : di servizio alla tabella “Specifiche_Opzioni”.

- **Cod_Tipo_Opz** (Numerico)
- Des_Tipo_OPz (Testo)

Tipo_Barriera_Opzione : di servizio alla tabella “Specifiche_Opzioni”.

- **Cod_Tipo_Barriera_Opz** (Numerico)
- Des_Tipo_Barriera_Opz (Testo)

Tipo_Contratto_Opzione : di servizio alla tabella “Specifiche_Opzioni”.

- **Cod_Tipo_Contratto_Opz** (Numerico)
- Des_Tipo_Cntratto_Opz (Testo)

Genere_Opzione : di servizio alla tabella “Specifiche_Opzioni”.

- **Cod_Genere_Opz** (Numerico)
- Des_Genere_Opz (Testo)

Sottotipo_Banking : di servizio alla tabella “Banking_BMT”.

- **Codice_Sottotipo** (Contatore)
- Sottotipo (Testo)

Tipi_Tassi : di servizio alla tabella “Tassi”.

- **Cod_Tipo_Tasso** (Contatore)
- Scadenza (Numerico)
- Tipo_Scadenza (Testo)

Sottotipo_Operazione : di servizio alla tabella “Operazioni”.

- **Codice_Sottotipo** (Contatore)
- Sottotipo_Operazione (Testo)

Tipologie_Mappatura : di servizio alla tabella “Mappatura_Fr”

- **Tipo_Mappatura** (Numerico)
- Des_Tipo_Mappatura (Testo)