

# Domande esame (possibili) + schema per esercitazione

**"Si descriva brevemente cosa siano indipendenza logica e fisica dai dati e quale/i modello di basi di dati consente/ono di realizzarle" (in alcune versioni: "...esplicitando quale caratteristica fondamentale del modello relazionale le consentano")**

L'architettura a livelli di un DBMS garantisce l'indipendenza dei dati in due livelli:

- **Indipendenza fisica:** consente di interagire con il DBMS in modo indipendente dalla struttura fisica dei dati, senza influire sulle descrizioni e quindi sui programmi che usano i dati
- **Indipendenza logica:** consente di interagire con il livello esterno della base di dati in modo indipendente dal livello logico, per esempio come aggiungere un nuovo schema esterno senza modificare lo schema logico e perciò la sottostante organizzazione fisica dei dati

Il **modello relazionale** risponde perfettamente al requisito dell'indipendenza dei dati perché introduce una netta distinzione tra il livello fisico e quello logico. A differenza dei modelli precedenti (reticolare e gerarchico), che includevano riferimenti esplicativi a puntatori fisici, il modello relazionale è basato su valori e non richiede la conoscenza delle strutture fisiche per accedere ai **dati**

**"Descrivere brevemente cosa sono le ACID properties, a cosa servono e da dove deriva l'espressione 'ACID'" oppure "Illustrare brevemente cosa siano le proprietà ACID (spiegando anche l'acronimo) ed a quale entità/oggetto sono riferite"**

Un DBMS prevede che le interazioni con la base di dati avvengano per mezzo di **transazioni**.

Una **transazione** è una sequenza di azioni di lettura e scrittura del DB e di elaborazioni di dati in memoria temporanea, che il DBMS esegue garantendo le seguenti proprietà (ACID properties):

- **Atomicity:** La transazione è eseguita nella sua interezza oppure non è eseguita affatto (le transazioni che terminano prematuramente sono abortite)
- **Consistency preservation:** una esecuzione corretta della transazione porta il DB da uno stato consistente all'altro (i vincoli di integrità devono essere rispettati).
- **Isolation:** L'esecuzione di una transazione deve essere indipendente da quella di altre transazioni concorrenti. Il risultato deve essere analogo a quello che si otterrebbe eseguendo le transazioni una alla volta (serialmente).

- **Durability** : le modifiche su DB di una transazione terminata normalmente sono permanenti, cioè non sono alterabili da malfunzionamenti successivi alla terminazione

## Si definisca brevemente cosa siano i trigger, il paradigma su cui sono basati, vantaggi e svantaggi per essi

I trigger, detti anche regole attive, sono dei costrutti per rendere la base di dati in grado di reagire a eventi definiti dall'amministratore tramite l'esecuzione di opportune azioni.

Seguono il paradigma Evento-Condizione-Azione (ECA), ossia:

- **Evento**: Tipicamente una modifica dello stato del database consiste in una operazione di INSERT, DELETE, UPDATE. Se l'evento accade, il trigger si attiva.
- **Condizione**: Predicato booleano espresso in SQL che identifica se l'azione del trigger deve essere eseguita. Quando il trigger si attiva, viene valutata la condizione, quindi il trigger viene considerato.
- **Azione**: Consiste in una sequenza di update SQL o una procedura. Quando la condizione è verificata, allora l'azione viene eseguita, quindi il trigger viene eseguito.

I trigger sono uno strumento molto potente che permette di gestire vincoli di integrità, calcolare dati derivati, gestire eccezioni e codificare regole aziendali. Un ulteriore vantaggio derivante dall'utilizzo dei trigger consiste nel riuscire a codificare la logica del sistema in maniera centralizzata e condivisa da tutte le applicazioni, con conseguenti vantaggi in fase di lettura e manutenzione del codice, infatti in caso di modifiche al comportamento del sistema è sufficiente intervenire nell'ambito della definizione dei trigger e non in più parti del codice.

Lo svantaggio è che i trigger sono standardizzati solo per SQL-3, per cui potrebbero presentarsi casi (se pur sempre più rari) di non portabilità del codice.

## Illustrare brevemente cosa sia il conflitto di impedenza e quali soluzioni esistano per gestirlo

Un importante problema che caratterizza l'integrazione tra SQL e i normali linguaggi di programmazione è il cosiddetto **conflitto di impedenza**. I linguaggi di programmazione accedono agli elementi di una tabella scandendone le righe una a una (tuple-oriented). Al contrario SQL è un linguaggio di tipo set-oriented, che opera su intere tabelle e restituisce come risultato di un'interrogazione un'intera tabella. Le soluzioni a questo problema si ottengono con l'utilizzo dei **cursori** e l'utilizzo di linguaggi con costruttori di tipo in grado di gestire una struttura del tipo "insieme di righe" (Call Level Interface).

## Si definiscano brevemente le nozioni di sistema organizzativo, sistema informativo e sistema informatico e si delinei la differenza esistente tra di essi

Un **sistema organizzativo** è un insieme di risorse e regole che consentono il funzionamento di una qualunque struttura sociale per il raggiungimento dei suoi obiettivi

Ogni sistema organizzativo è dotato di **sistema informativo**, ossia l'insieme delle risorse e delle procedure che un'organizzazione utilizza per gestire le informazioni necessarie al perseguitamento dei propri scopi.

Un sistema informativo è molte volte indipendente dalle automatizzazioni che conosciamo.

Un **sistema informatico** non è altro che la parte automatizzata di un sistema informativo, quindi un sistema software orientato alla gestione dei dati, dove l'aspetto prevalente è rappresentato dai dati stessi (memorizzati, ricercati, modificati) che costituiscono il patrimonio informativo di un'**organizzazione**

## **Si descriva brevemente i/il criteri/o di ottimizzazione delle query implementato dal gestore di ottimizzazione delle query di un DBMS**

Il gestore delle interrogazioni è un modulo cruciale dell'architettura di un DBMS, in quanto responsabile dell'esecuzione efficiente di operazioni che sono specificate a livello molto alto. Esso riceve in ingresso un'interrogazione scritta in SQL, controlla che non vi siano errori lessicali, sintattici o semantici, una volta accettata, l'interrogazione viene tradotta in una forma interna di tipo algebrico. A questo punto, l'ottimizzazione vera e propria ha inizio, dividendosi in:

1. **Ottimizzazione algebrica**: effettua trasformazioni sulle operazioni (come l'anticipazione di selezioni e proiezioni verso le foglie dell'albero) che sono sempre convenienti indipendentemente dai costi fisici
2. **Ottimizzazione basata sul modello dei costi**: Sceglie la strategia di esecuzione (es. quale indice usare, quale algoritmo di join tra nested loop, merge scan o hash join) basandosi su un modello di costo che stima il numero di accessi in memoria secondaria e l'uso della CPU, sfruttando i profili statistici delle tabelle memorizzati nel catalogo (come spiegato nei profili delle relazioni);
3. Generazione del codice.

## **Elencare ed illustrare brevemente le strutture usate da un DBMS per organizzare i file (e le strutture primarie dei file) a livello fisico**

La struttura primaria di un file stabilisce il criterio secondo il quale sono disposte le tuple nell'ambito del file. Le strutture possono essere divise in tre categorie principali:

- Sequenziali;
- Ad accesso calcolato (hash);
- Ad albero;

Nelle strutture sequenziali, un file è costituito da vari blocchi di memoria "logicamente" consecutivi, e le tuple vengono inserite nei blocchi rispettando una sequenza:

- **Seriale**: sequenza delle tuple indotta dall'ordine di immissione (organizzazione disordinata).

Di solito viene chiamata anche **heap**, ossia mucchio

- **Array**: le tuple sono disposte come in un array, e la loro posizione dipende dal valore assunto in ciascuna tupla da un campo di indice.  
Possibile soltanto quando le tuple di una tabella sono di dimensione fissa.
- **Ordinata**: la sequenza delle tuple dipende dal valore assunto in ciascuna tupla da un campo (attributo) del file, ossia la chiave.

Una struttura con accesso ad **hash** garantisce un accesso associativo ai dati, ovvero un tipo di accesso in cui la locazione fisica dei dati dipende dal valore assunto da un campo chiave. Questo avviene tramite specifiche funzioni hash che consentono di trasformare un attributo chiave nell'indice di un array, e quindi associare ad ogni record una posizione specifica in una struttura sequenziale.

Le strutture ad albero, denominate anche **indici**, favoriscono l'accesso in base al valore di uno o più campi, consentendo sia accessi puntuali che corrispondenti a valori con complessità logaritmica (sulla base della profondità dell'albero).

## **Definire cosa siano RDF ed RDFS e le differenze tra di essi" oppure "Descrivere brevemente RDF e quale sia il suo linguaggio di interrogazione**

**RDF (Resource Description Framework)**: È un modello per rappresentare informazioni sul Web sotto forma di **triple** (soggetto, predicato, oggetto), che formano un grafo orientato. Permette di descrivere risorse in modo semplice ma non impone uno schema rigido

**RDFS (RDF Schema)**: È un'estensione di RDF che permette di definire **metadati** (uno schema). Introduce concetti come **classi**, **proprietà** e relazioni di ereditarietà (*subClassOf*), consentendo di descrivere la struttura e derivare nuova conoscenza (inferenza). Il linguaggio di interrogazione standard per RDF è **SPARQL**, uno standard per interrogare una o più istanze RDF, restituendo come risultato un file XML.

**SPARQL** introduce, oltre alle forme classiche di interrogazione presenti nell'SQL, altri costrutti come:

- **DESCRIBE** per ottenere una descrizione delle risorse presenti presso un endpoint
- **ASK** per sapere se specifici termini sono disponibili nell'endpoint
- **COSTRUCT** per costruire un nuovo grafo RDF a partire da un'interrogazione

## **Illustrare brevemente cosa siano B-tree e B+-Tree e le differenze tra essi" oppure "Illustrare brevemente cosa sia un B+-Tree..." o "Determinare quali azioni possono essere necessarie su una struttura dati di tipo B-Tree.**

Le strutture ad albero, denominate anche **indici**, favoriscono l'accesso in base al valore di uno o più campi, consentendo sia accessi puntuali che corrispondenti a valori con

complessità logaritmica (sulla base della profondità dell'albero).

Queste strutture sono usate per l'indicizzazione. Garantiscono che le foglie siano tutte alla stessa distanza dalla radice, offrendo tempi di accesso logaritmici

Le differenze principali sono:

- **B-Tree:** I nodi intermedi possono contenere i dati veri e propri (o i puntatori ai record).
- **B+-Tree:** I dati (o i puntatori ai record) sono contenuti **solo nelle foglie**. I nodi interni servono solo da instradamento. Inoltre, le foglie sono collegate in una catena sequenziale, rendendo questa struttura molto efficiente anche per interrogazioni su intervalli di valori

## **Descrivere brevemente cosa è la normalizzazione e quali problemi risolve**

Esistono alcune proprietà, dette **forme normali**, che certificano la qualità dello schema di una base di dati relazionale tramite l'assenza di determinati difetti.

Quando una relazione non è normalizzata presenta ridondanze e si presta a comportamenti indesiderabili o anomali durante gli aggiornamenti.

Dunque, per **normalizzazione** si intende la procedura che permette di trasformare schemi non normalizzati in schemi che soddisfano una forma normale.

## **Si definisca cosa è una decomposizione senza perdita di informazione e quale condizione è possibile utilizzare per verificare tale proprietà**

Data una relazione  $r$  su un insieme di attributi  $X$ , con  $X_1$  e  $X_2$  sottoinsiemi di  $X$  la cui unione sia pari a  $X$  stesso, si può decomporre senza perdita di dati sugli insiemi  $X_1$  e  $X_2$  se il join delle due proiezioni è uguale a  $r$  stessa (ossia non contiene **spurie**).

Quindi si decompone senza perdita su due sottoschemi se l'attributo comune ai due è chiave per almeno uno dei due

## **Si definisca cosa è un vincolo di integrità per una base di dati relazionale**

Il **vincolo di integrità** è una proprietà che deve essere soddisfatta dalle istanze che rappresentano informazioni corrette per l'applicazione.

Ogni vincolo può essere visto come un predicato che assegna valori vero o falso se queste soddisfano le condizioni o no.

## **Si definisca cosa sia una superchiave ed una superchiave minimale (o chiave).**

Una **chiave** è un insieme di attributi utilizzato per identificare univocamente le tuple di una relazione, formalmente:

**Def.:** Data una relazione  $R(X)$  con  $K \subseteq X$ , l'insieme  $K$  di attributi è superchiave per  $R$  se  $r$  non contiene due tuple distinte  $t_1$  e  $t_2$  con  $t_1[K] = t_2[K]$ .

In altre parole l'insieme di istanze di attributi utilizzato come superchiave non deve contenere elementi uguali in più di una tupla.

Un insieme  $K$  di attributi è chiave per  $r$  se è una superchiave minimale di  $r$  (cioè non esiste un'altra superchiave  $K'$  di  $r$  che sia contenuta in  $K$  come sottoinsieme proprio).

Poiché una relazione è un insieme di elementi distinti, si può concludere che per ogni relazione  $r(X)$ , l'insieme  $X$  di tutti gli attributi su cui è definita è chiaramente una superchiave per essa.

Ora i casi sono due:

- La **superchiave è anche chiave**, quindi si conferma l'esistenza della chiave stessa
- **Non è chiave**, perché contiene un'altra superchiave, quindi applicando ricorsivamente questo ragionamento si giunge, in un numero finito di passi (poiché l'insieme degli attributi è finito), ad una superchiave minimale.

## Elencare le proprietà fondamentali di una transazione e descrivere brevemente ognuna di esse

Una **transazione** è una sequenza di azioni di lettura e scrittura del DB e di elaborazioni di dati in memoria temporanea, che il DBMS esegue garantendo le seguenti proprietà (ACID properties):

- **Atomicity**: La transazione è eseguita nella sua interezza oppure non è eseguita affatto (le transazioni che terminano prematuramente sono abortite)
- **Consistency preservation**: una esecuzione corretta della transazione porta il DB da uno stato consistente all'altro (i vincoli di integrità devono essere rispettati).
- **Isolation**: L'esecuzione di una transazione deve essere indipendente da quella di altre transazioni concorrenti. Il risultato deve essere analogo a quello che si otterrebbe eseguendo le transazioni una alla volta (serialmente).
- **Durability** : le modifiche su DB di una transazione terminata normalmente sono permanenti, cioè non sono alterabili da malfunzionamenti successivi alla terminazione

## Si descrivano brevemente i diversi livelli di isolamento di una transazione ed il motivo per cui sono stati introdotti.

1. **Read uncommitted (degree of isolation 0)**: consente transazioni che fanno solo operazioni di lettura (quelle di modifica sono proibite) che vengono eseguite dal sistema senza bloccare in lettura i dati. Si rende il sistema molto più veloce, ma può accadere che una transazione legga dati modificati da un'altra transazione non ancora terminata (dati sporchi) oppure abortita in seguito, motivo per cui questo livello di isolamento può applicarsi esclusivamente su porzioni di DB utilizzate sempre e solo in lettura.

2. **Read committed (degree of isolation 1):** a differenza del livello precedente in cui sui dati in lettura non vi era un bloccaggio, questo livello prevede che i dati in lettura siano bloccati esclusivamente per il tempo di lettura e subito rilasciati, mentre i dati in scrittura siano rilasciati alla terminazione della transazione. Questo comporta letture non ripetibili, ovvero letture successive sugli stessi dati possono dare risultati diversi perché i dati sono stati modificati da altre transazioni terminate nell'intervallo tra la prima e la seconda lettura.
3. **Repeatable read (degree of isolation 2):** prevede che i blocchi in lettura e scrittura non sia applicati sull'intera tabella, ma siano assegnati solo su sottoinsiemi di tuple e vengano rilasciati alla terminazione della transazione. Questa soluzione evita il problema delle letture non ripetibili, ma non quello delle letture fantasma.
4. **Serializable (degree of isolation 4):** le transazioni vengono serializzate in maniera sicura.

## **Illustrare brevemente quali sono i dati gestiti dal buffer manager e come avviene la sua gestione delle richieste**

Il gestore del buffer gestisce, oltre al buffer appunto, un direttorio che per ogni pagina mantiene:

- File fisico e numero blocco corrispondente alla pagina
- Due variabili di stato: un contatore che indica quanti programmi utilizzano la pagina, un bit che indica se la pagina è “sporca”, cioè se è stata modificata

La conoscenza di questi dati è fondamentale nel momento in cui diventa necessario introdurre nuove pagine in un buffer saturo, per capire quali pagine andare a sostituire. Il buffer comunica con il sistema mediante delle operazioni primitive:

- fix: consiste nella richiesta di accesso ad una pagina, restituisce il riferimento alla pagina richiesta, richiede una lettura se la pagina non è nel buffer, incrementa il contatore per l'utilizzo della pagina
- setDirty: comunica al buffer manager che la pagina è stata modificata
- unfix: indica che la transazione ha concluso l'utilizzo della pagina, quindi decrementa il contatore di utilizzo di pagina
- force: trasferisce in modo sincrono una pagina in memoria secondaria su richiesta del gestore dell'affidabilità.

## **E' possibile creare domini complessi in SQL? Quali sono i domini che SQL mette a disposizione?**

Nella definizione delle tabelle si può fare riferimento ai domini predefiniti del linguaggio o a domini definiti dall'utente a partire da quelli predefiniti, infatti proprio da questi è possibile definirli in questa maniera:

Domande esame (possibili) + schema per esercitazione  
CREATE DOMAIN *NomeDominio* as *TipoDiDato*  
[*ValoreDiDefault*]  
[*Vincolo*]

I domini elementari di SQL sono:

- Caratteri
- Tipi numerici esatti
- Tipi numerici approssimativi
- Istanti temporali
- Intervalli temporali

## **Descrivere brevemente in cosa consiste SQL Embedded e per quale motivo viene introdotto**

SQL Embedded prevede di introdurre direttamente nel programma sorgente scritto nel linguaggio di alto livello le istruzioni SQL, distinguendole dalle normali istruzioni tramite un opportuno separatore

## **Si descriva brevemente cosa sia un cursore e per risolvere quale problema viene introdotto**

Un cursore è una variabile speciale che permette ad un programma di accedere alle righe di una tabella una alla volta e permette di risolvere il cosiddetto **conflitto di impedenza**, infatti linguaggi di programmazione accedono agli elementi di una tabella scandendone le righe una a una (tuple-oriented), al contrario SQL è un linguaggio di tipo set-oriented, che opera su intere tabelle e restituisce come risultato di un'interrogazione un'intera tabella.

## **Descrivere le caratteristiche principali di una procedura definita in SQL-2 standard e le eventuali differenze rispetto a SQL-3**

Lo standard SQL-2 prevede la definizione di **Procedure**, ovvero dei brevi sottoprogrammi memorizzati nel database come parte dello schema (motivo per cui vengono dette anche **stored procedures**). Esse permettono di assegnare un nome a un'istruzione SQL ed eventuali parametri. Una volta definita, la procedura è utilizzabile come un qualunque comando SQL.

È bene sapere che lo standard SQL-2 non tratta la scrittura di procedure complesse, ma solo quelle composte da un singolo comando SQL. Questo è invece permesso in SQL-3, dove viene fornita una ricca sintassi per la definizione di procedure, integrando anche le strutture di controllo

## **Illustrare quando è possibile utilizzare SQL statico e quando invece è necessario utilizzare SQL dinamico.**

Nel caso di SQL statico, si usa quando sono noti a tempo di compilazione e vengono gestiti dal preprocessore, venendo ottimizzati solo una volta, e non ogni volta che il comando deve essere eseguito. Questo comporta grossi vantaggi in termini di prestazioni. L'SQL dinamico non può avvalersi della fase di preprocessamento, non essendo noti a priori i comandi da ottimizzare, quindi la costruisce a tempo di esecuzione, quindi viene utilizzato in questi casi necessari.

## **Illustrare brevemente cosa siano: algebra relazionale, calcolo relazionale ed SQL, le loro peculiarità e la relazione che intercorre tra di essi**

L'algebra relazionale si configura come un linguaggio di tipo procedurale che si basa su una collezione di operatori definiti su relazioni, i quali producono a loro volta nuove relazioni come risultato. La sua peculiarità fondamentale risiede nella necessità di specificare esplicitamente il procedimento da seguire, ovvero la sequenza di operazioni quali selezione  $\sigma$ , proiezione  $\pi$ , ridenominazione  $\rho$  e diverse forme di join  $\bowtie$ , per giungere alla costruzione del risultato desiderato.

Il calcolo relazionale costituisce una famiglia di linguaggi di interrogazione di natura dichiarativa, basati sui principi del calcolo dei predicati del primo ordine. La caratteristica distintiva del calcolo è la sua capacità di descrivere esclusivamente le proprietà del risultato cercato attraverso formule logiche, senza indicare minimamente la procedura necessaria per ottenerlo .

Il linguaggio SQL, acronimo di Structured Query Language, rappresenta lo standard universale per l'interazione con le basi di dati relazionali, integrando in un'unica soluzione le funzionalità di definizione dei dati attraverso il Data Definition Language e di manipolazione e interrogazione tramite il Data Manipulation Language. La peculiarità dell'SQL è la sua impostazione dichiarativa che solleva l'utente dal compito di definire il piano di accesso fisico, delegando al sistema la ricerca della strategia ottimale

La relazione che intercorre tra questi linguaggi è di natura formale e tecnologica, in quanto l'algebra relazionale e il sottoinsieme del calcolo relazionale indipendente dal dominio sono considerati linguaggi equivalenti, poiché per ogni espressione formulata nell'uno è possibile trovarne una corrispondente nell'altro che produca il medesimo insieme di dati.

Nella pratica dei sistemi di gestione di basi di dati, l'SQL funge da interfaccia di alto livello basata sul calcolo su tuple, ma il gestore delle interrogazioni del DBMS traduce internamente ogni istruzione SQL in una rappresentazione algebrica equivalente

## **Descrivere brevemente i difetti del calcolo relazionale su domini per i quali è stato introdotto il calcolo relazionale su tuple**

Il calcolo relazionale su domini presenta alcuni limiti strutturali e teorici piuttosto significativi, tra i quali il principale è rappresentato dalla dipendenza dal dominio, una proprietà per cui il risultato di un'interrogazione può variare in base all'universo dei valori considerato per gli attributi. Questo difetto implica che alcune espressioni possano produrre risultati privi di

senso pratico o addirittura infiniti qualora il dominio di riferimento sia illimitato, come accade ad esempio in una negazione che includa tutti i valori non presenti in una determinata relazione

Per correggere queste problematiche è stato introdotto il calcolo relazionale su tuple con dichiarazioni di range, il quale apporta un cambiamento fondamentale definendo variabili che denotano intere tuple invece di singoli valori atomici

## **Si definisca brevemente cosa sia un DBMS e le sue principali caratteristiche**

Il DBMS (sistemi di gestioni di basi di dati) è un sistema software in grado di gestire collezioni di dati che siano grandi, condivise e persistenti, garantendo affidabilità, privatezza, efficienza ed efficacia.

Un DBMS deve garantire:

- **Affidabilità**, quindi una resistenza a malfunzionamenti HW e SW in modo da mantenere intatto il contenuto o permetterne la ricostruzione (backup e recovery).
- **Privatezza dei dati**, un sistema deve poter definire dei meccanismi di autorizzazione per utente (opportunamente riconosciuto)

## **Data una base di dati si dica quale sia la sua parte invariante e la sua parte variabile e perché**

Nella base di dati esiste una parte invariata nel tempo, detta **schema della base di dati**, costituita dalle caratteristiche dei dati, e una parte variabile, chiamata **istanza** della base di dati, costituita dai valori effettivi.

## **Descrivere brevemente cosa sia un modello dei dati ed elencare i modelli dei dati conosciuti**

Un **modello di dati** è un insieme di concetti (o costrutti) per organizzare i dati di interesse e descriverne la struttura in modo comprensibile ad un elaboratore.

Ogni modello di dati fornisce meccanismi di astrazione per definire nuovi tipi sulla base di tipi (elementari) predefiniti e costruttori di tipo.

Il modello relazione dei dati (più diffuso tra tutti) permette di definire tipi per mezzo del costrutto della **relazione**, che consente di organizzare i dati in insiemi di record a struttura fissa. Oltre quello esistono diversi

Modello	Struttura usata	Anni	DBMS
Modello gerarchico	basato sull'uso di strutture ad albero	'60	IMS System 2000

Modello	Struttura usata	Anni	DBMS
Modello reticolare	basato sull'uso di strutture a grafo	Inizio '70	IDMS, IDS II, DM IV
Modello relazionale	basato sull'uso di relazioni: insiemi di record a struttura fissa con campi di tipo primitivo	'80	System R, Ingres, Oracle, DB2
Modello ad oggetti	basato sull'uso di classi di oggetti e istanze	Fine '80 - '90	O2 ObjectStore
Modello XML	rivisitazione del modello gerarchico: dati presentati insieme alla loro descrizione e non devono sottostare rigidamente ad un'unica struttura logica	'90	BaseX
Modelli semi-strutturati e flessibili	non hanno rigidità nell'organizzazione dei dati ed hanno alte prestazioni	'00	Sistemi NoSQL

1. Progettazione Concettuale: La finalità di questa fase è rappresentare le specifiche informali della realtà di interesse in una **descrizione formale e completa**, ma totalmente **indipendente** dai criteri di rappresentazione del DBMS che verrà utilizzato.
  - **Peculiarità:** Si caratterizza per un **alto livello di astrazione**. Il progettista si concentra esclusivamente sul contenuto informativo della base di dati, senza preoccuparsi delle modalità di codifica in un sistema reale né dell'efficienza dei programmi. In questa fase si utilizzano principalmente le specifiche sui dati per definire gli elementi dello schema.
  - **Prodotto finale:** Il risultato è lo **schema concettuale**, tipicamente realizzato attraverso il **modello Entità-Relazione (E-R)**, che fornisce una visione unificata dei dati utile anche a scopo documentativo.
2. Progettazione Logica: La finalità della progettazione logica è la **traduzione dello schema concettuale** nel modello di rappresentazione dei dati adottato dal DBMS a disposizione (nel caso specifico, il modello relazionale).
  - **Peculiarità:** Sebbene sia ancora indipendente dai dettagli fisici, la rappresentazione è **concreta** poiché deve rispettare i vincoli del modello logico scelto. Non si tratta di una semplice traduzione meccanica: lo schema deve essere **ristrutturato** per semplificare la traduzione stessa (ad esempio eliminando le generalizzazioni) e per **ottimizzare le prestazioni** sulla base del carico applicativo previsto (volumi di dati e frequenza delle operazioni). In questa fase si utilizzano tecniche formali come la **normalizzazione** per verificare la qualità del risultato
  - **Prodotto finale:** Il risultato è lo **schema logico** (una collezione di tabelle nel modello relazionale) e la relativa documentazione dei vincoli di integrità.
3. Progettazione Fisica: La finalità della progettazione fisica è il completamento dello schema logico con la specifica dei **parametri fisici di memorizzazione** per

massimizzare l'efficienza del sistema.

- **Peculiarità:** Questa fase è **fortemente dipendente dal DBMS specifico** scelto. Il progettista deve conoscere le caratteristiche tecnologiche del sistema per definire l'organizzazione dei file e la creazione di strutture ausiliarie per l'accesso ai dati. Qui si prendono decisioni su quali indici creare per ottimizzare i tempi di risposta delle query e sulla contiguità di allocazione dei dati.
- **Prodotto finale:** Il risultato è lo **schema fisico**, costituito dalle definizioni effettive delle relazioni e delle strutture fisiche utilizzate con i relativi parametri di implementazione.

## **Si descriva brevemente quali e quante sono le forme di ridondanza individuabili all'interno di uno modello E-R**

All'interno di uno schema Entità-Relazione, la ridondanza è definita come la presenza di informazioni che possono essere derivate, ovvero ottenute attraverso una serie di operazioni, da altri dati già esistenti nel medesimo schema. Si possono individuare sostanzialmente tre forme principali di ridondanza:

1. La prima categoria riguarda gli attributi derivabili, occorrenza per occorrenza, da altri attributi appartenenti alla stessa entità o alla stessa associazione.
2. La seconda forma di ridondanza è costituita dagli attributi derivabili da attributi di altre entità o associazioni, operazione che avviene solitamente attraverso l'impiego di funzioni aggregate;
3. La terza forma riguarda le associazioni derivabili dalla composizione di altre associazioni in presenza di cicli nello schema.

## **Illustrare brevemente i motivi per cui è necessario effettuare l'analisi della ridondanza**

La presenza di una ridondanza ha effetti positivi, semplificare le interrogazioni, ed effetti negativi, appesantisce gli aggiornamenti e comporta l'occupazione di più memoria. Per questo motivo, la decisione di mantenere o eliminare una ridondanza va presa in seguito ad una **analisi quantitativa** che confronti il costo di esecuzione delle operazioni che coinvolgono il dato ridondante e l'occupazione di memoria, sia in presenza e che in assenza di ridondanza.

## **Si descriva brevemente quando un join naturale si dice completo**

Si parla di **join completo** se ogni tupla di ciascun operando contribuisce ad almeno una tupla del risultato.

## **Descrivere brevemente perché il modello relazionale è anche detto modello 'basato su valori'**

Il modello relazionale viene definito basato su valori poiché, a differenza dei modelli logici precedenti come quello gerarchico o reticolare, le corrispondenze e i riferimenti tra i dati contenuti in relazioni diverse vengono rappresentati esclusivamente attraverso valori comuni che compaiono nelle tuple, senza l'impiego di puntatori esplicativi o riferimenti a indirizzi fisici.

---

## Bullet list per svolgere tracce d'esame

Questa è una piccola lista basata sulle regole generali date dalla prof per svolgere l'esame, visto che è impossibile trovare informazioni a quanto vi si è capito.

**IMPORTANTE:** ogni punto deve essere esplicitato nell'esame, oltre ad avere una maniera molto più discorsiva di quella descritta qui. Tutti i punti che non vengono toccati (come può essere l'analisi delle ridondanze) deve essere argomentato sul perché non venga fatto

## Richieste solite

Nelle tracce viene chiesto di eseguire:

- **Analisi dei requisiti**
- **Progettazione concettuale**
- **Progettazione logica**

del DB necessario alla realizzazione di un tale sistema informatico, specificando la strategia di progetto scelta e descrivendola brevemente.

Si specifichino tutti i passi di cui si compongono analisi dei requisiti, progettazione concettuale e progettazione logica e si scelga quali di essi eseguire, motivando sia le azioni dei passi eseguiti sia il perché taluni passi non sono eventualmente esplicitati (es.: non si riporta la linearizzazione delle frasi perché le frasi sono sufficientemente linearizzate).

In alcune tracce sono presenti anche questi due quesiti:

- Determinare a propria scelta tavola degli accessi e delle operazioni quando necessario ed eventuali attributi ritenuti utili.
- Al termine della progettazione logica, **stabilire se le relazioni ottenute sono in forma normale (se si quale) o meno, giustificando la risposta secondo la teoria**

Lo schema generale di azione, in ordine, è il seguente:

1. Analisi dei requisiti
  - Set di regole
  - Riorganizzazione per concetti
  - Creazione del glossario
  - Specifiche sui dati e operazioni
2. Progettazione concettuale

- Scelta della strategia
- Schema scheletro
- Raffinamenti
- Vincoli e valori ammessi

### 3. Progettazione logica

- Ristrutturazione
- Determinazione carico applicativo (ossia tavola dei volumi e delle operazioni)
- Determinazione costo singola operazione
- Traduzione modello relazionale

## Analisi dei requisiti

Questa fase prevede la riscrittura della traccia in una maniera più specifica.

Noi faremo riferimento a questa traccia in particolare:

### Esercizio

*Si intende automatizzare il sistema di gestione di una raccolta storica di fotografie ad opera di un ente. In seguito alla raccolta dei requisiti si è ottenuto quanto segue:*

Le fotografie sono conservate in un archivio distribuito su varie sedi. Per ogni sede si conosce: il responsabile dell'archivio per quella sede, l'indirizzo, il numero telefonico, l'orario di apertura e l'orario di chiusura. Le foto sono catalogate rispetto ad un catalogo di soggetti possibili, ognuno dei quali ha un proprio identificativo. Le foto possono descrivere personaggi, luoghi o oggetti e possono essere in bianco e nero o a colori. Ogni foto ha una propria dimensione ed un proprio stato di conservazione; per le foto a colori si conosce anche il tipo di stampa (chiaro o opaco). Nel caso in cui il soggetto sia un personaggio, si conoscono: nome, sesso, se in vita o deceduto. Nel caso di personaggi politici, si dispone anche delle informazioni relative a: partito di appartenenza, eventuale carica governativa ricoperta. Nel caso di artisti, si tiene traccia del nome e dell'attività prevalente (es. pittura, scultura, etc...). Nel caso di foto che descrivano luoghi o oggetti, si conoscono: nome e descrizione. Se l'oggetto è un'opera artistica, si conoscono anche: il nome dell'opera d'arte, l'artista che l'ha realizzata, il luogo dove l'opera risiede e l'anno di realizzazione.

Il sistema deve essere in grado di gestire, tra le altre, le seguenti operazioni:

- 1)visualizzare l'insieme delle fotografie per un certo soggetto e la loro dislocazione fisica
- 2)inserire un nuovo soggetto nel catalogo
- 3)inserire una nuova fotografia
- 4)modificare i dati per una sede (es. il responsabile, il numero di telefono, l'orario di apertura e/o chiusura)
- 5)visualizzare le informazioni per una specifica foto
- 6)modificare le informazioni relative ad un personaggio

## Set di regole

- **Scegliere il corretto livello di astrazione:** evitare termini troppo astratti o troppo specifici  
**Esempio:** dimensioni di una foto (piccola, media, grande), specificare che non esistono determinati dati (come il giorno di chiusura per una sede)
- **Standardizzare la struttura delle frasi,** ossia usare sempre lo stesso stile sintattico  
**Esempio:** Per < dato > si rappresenta < proprietà >
- **Linearizzare le frasi e suddividere quelle articolate**  
**Esempio:** Preferibile dipendenti statali piuttosto che "quelli che lavorano per un ente dello stato"
- **Rendere esplicito il riferimento fra termini,** ossia riconoscere quale sia la giusta relazione tra i termini (per esempio se un numero di telefono appartiene ad un responsabile o a qualcun'altro)
- **Individuare omonimi e sinonimi,** ossia unificare i termini  
**Esempio:**  
Fotografia e Foto sono sinonimi → si sceglie **Foto**  
Opera artistica e opera d'arte → si sceglie **Opera d'arte**

## Riorganizzazione di concetti

In questa fase riorganizziamo i vari concetti per frase e per entità rilevata, nel nostro caso:

- **FRASI DI CARATTERE GENERALE**

Si intende automatizzare il sistema di gestione di una raccolta storica di foto ad opera di un ente. Le foto sono conservate in un archivio distribuito su varie sedi.

- **FRASI RELATIVE ALLA SEDE**

Le foto sono conservate in un archivio distribuito su varie sedi. Per ogni sede si conosce: il responsabile **sede archivio**, indirizzo, numero telefonico, orario di apertura e orario di chiusura (**non sono previsti giorni di chiusura**). Del responsabile sede archivio si conoscono nome, cognome e codice fiscale.

## • **FRASI RELATIVE ALLE FOTO**

Le foto sono conservate in un archivio distribuito su varie sedi. Le foto sono catalogate rispetto a soggetti possibili. Oltre al soggetto, per ogni foto si conoscono anche le seguenti informazioni: se è in bianco e nero o a colori, la dimensione (piccola, media, grande), lo stato di conservazione (scarsa, sufficiente, buono, ottimo). Nel caso di foto a colori si conosce anche il tipo di stampa (chiaro o opaco).

## • **FRASI RELATIVE AI SOGGETTI**

Le foto sono catalogate rispetto a soggetti possibili. Ogni soggetto ha un identificativo. I soggetti possibili sono: personaggi, luoghi o oggetti.

### • **FRASI RELATIVE AL SOG. “PERSONAGGIO”**

Nel caso il soggetto di una foto sia un personaggio, del personaggio si conoscono: nome, sesso, se in vita o deceduto. Nel caso di personaggi politici, si conoscono anche: partito di appartenenza, carica governativa ricoperta (opzionale). Nel caso di personaggi artisti, si conoscono: nome e attività prevalente (es. pittura, scultura, ...).

### • **FRASI RELATIVE AL TIPO SOGGETTO. “LUOGO”**

Nel caso il soggetto di una foto sia luogo, si conoscono: nome del luogo e descrizione.

### • **FRASI RELATIVE AL TIPO SOG. “OGGETTO”**

Nel caso il soggetto di una foto sia un oggetto, si conoscono: nome dell'oggetto e descrizione. Se l'oggetto è un'opera d'arte, si conoscono anche: il nome dell'opera d'arte, il nome e cognome dell'artista che l'ha realizzata, il luogo dove l'opera risiede e l'anno di

- **Costruire un glossario per termini**

Termino	Descrizione	Sinonimi	Collegamenti
Sede	Luogo fisico che ospita un sottousieme di foto. Ci sono diverse sedi.	-----	Foto
Foto	Elemento oggetto dell'archiviazione	Fotografia	Sede, Soggetto
Soggetto	Immagine principale rappresentata dalla foto. Ci possono essere diverse tipologie di soggetto. Le foto sono catalogate rispetto ad un solo soggetto.	-----	Foto, Personaggio, Oggetto, Luogo
Personaggio	Una delle tipologie di soggetto risp. cui catalogare le foto	-----	Soggetto
Luogo	Una delle tipologie di soggetto risp. cui catalogare le foto	-----	Soggetto
Oggetto	Una delle tipologie di soggetto risp. cui catalogare le foto	-----	Soggetto

## Specifiche delle operazioni

In questa fase bisogna prendere le operazioni specificate dalla traccia e dargli dei valori possibili per il database (utili nelle parti successive)

Il sistema deve essere in grado di gestire, tra le altre, le seguenti operazioni:

- 1) visualizzare l'insieme delle fotografie per un certo **tipo di** soggetto e la loro dislocazione fisica (50 volte al gg)
- 2) inserire un nuovo tipo di soggetto nel catalogo (10 volte l'anno)
- 3) Inserire una nuova fotografia (100 volte a settimana)
- 4) modificare i dati per una sede (es. il responsabile, il numero di telefono, l'orario di apertura e/o chiusura) (20 volte l'anno)
- 5) visualizzare le informazioni per una specifica foto (200 volte al giorno)
- 6) modificare le informazioni relative ad un personaggio (10 volte al mese)

## Progettazione concettuale

### Scelta della strategia

Qui andiamo a descrivere la strategia scelta per il nostro schema. Nel 99% dei casi conviene la strategia ibrida, ossia:

- Partendo dalle specifiche si rappresentano tutte le informazioni in uno schema scheletro iniziale usando pochi concetti astratti.
- Successivamente ogni entità/relazione dello schema è raffinata rispetto al documento di specifica dei requisiti finché tutte le specifiche non sono rappresentate
- I diversi schemi ottenuti sono integrati, sulla base delle associazioni indicate nello schema scheletro eventualmente raffinate, giungendo allo schema E-R finale, più dettagliato di quello iniziale.

## Schema scheletro

Andiamo a costruire uno schema scheletro su cui baseremo i vari raffinamenti:



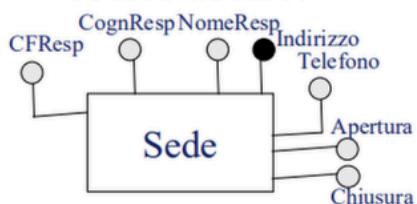
## Raffinamenti

Qui andiamo ad espandere lo schema scheletro in uno più completo, basandoci sull'analisi dei requisiti e in particolare alla [Riorganizzazione di concetti](#):

### FRASI RELATIVE ALLA SEDE

Le foto sono conservate in un archivio distribuito su varie sedi.

Per ogni sede si conosce: il responsabile **sede archivio**, indirizzo, numero telefonico, orario di apertura e orario di chiusura (non sono previsti giorni di chiusura). Del responsabile sede archivio si conoscono nome, cognome e codice fiscale.

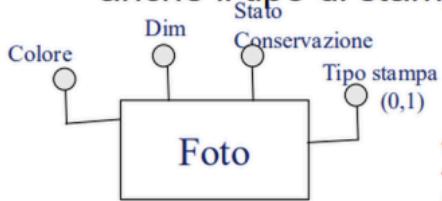


Una persona può essere responsabile di più di una sede contemporaneamente

### FRASI RELATIVE ALLE FOTO

Le foto sono conservate in un archivio distribuito su varie sedi.

Le foto sono catalogate rispetto a soggetti possibili. Oltre al soggetto, per ogni foto si conoscono anche le seguenti informazioni: se è in bianco e nero o a colori, la dimensione (piccola, media, grande), lo stato di conservazione (scarsa, sufficiente, buono, ottimo). Nel caso di foto a colori si conosce anche il tipo di stampa (chiaro o opaco).



Le foto sono catalogate rispetto ad un solo soggetto (soggetto dominante)

#### Valori Ammessi

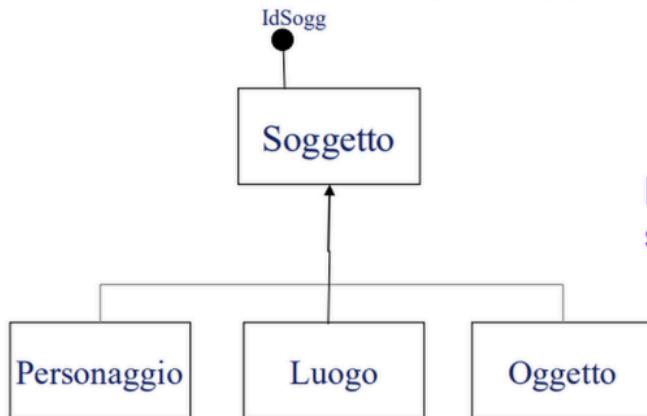
Colore = {B-N, colori} Dim = {picc, med, grande}  
Stato Conservazione = {scarsa, sufficiente, buono, ottimo} Tipo stampa = {chiaro, opaco}

#### Vincoli

“Tipo Stampa” è avvalorato solo se “Colore”=”colori”

## FRASI RELATIVE AI SOGGETTI

Le foto sono catalogate rispetto a soggetti possibili. Ogni soggetto ha un identificativo. I soggetti possibili sono: personaggi, luoghi o oggetti.

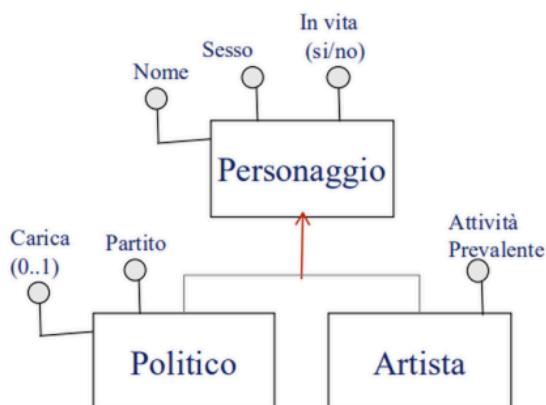


Le foto sono catalogate rispetto ad un solo soggetto (soggetto dominante)

## FRASI RELATIVE AL SOGGETTO “PERSONAGGIO”

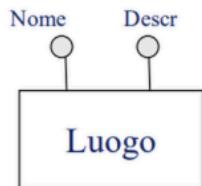
Nel caso il soggetto di una foto sia un personaggio, del personaggio si conoscono: nome, sesso, se in vita o deceduto.

Nel caso di personaggi politici, si conoscono anche: partito di appartenenza, carica governativa ricoperta (opzionale). Nel caso di personaggi artisti, si conoscono: nome e attività prevalente (es. pittura, scultura, ...).



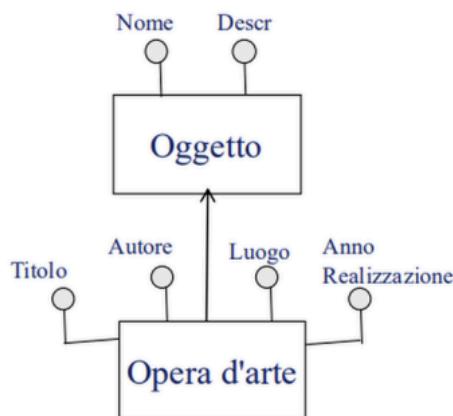
## FRASI RELATIVE AL SOGGETTO “LUOGO”

Nel caso il soggetto di una foto sia luogo, si conoscono: nome del luogo e descrizione.



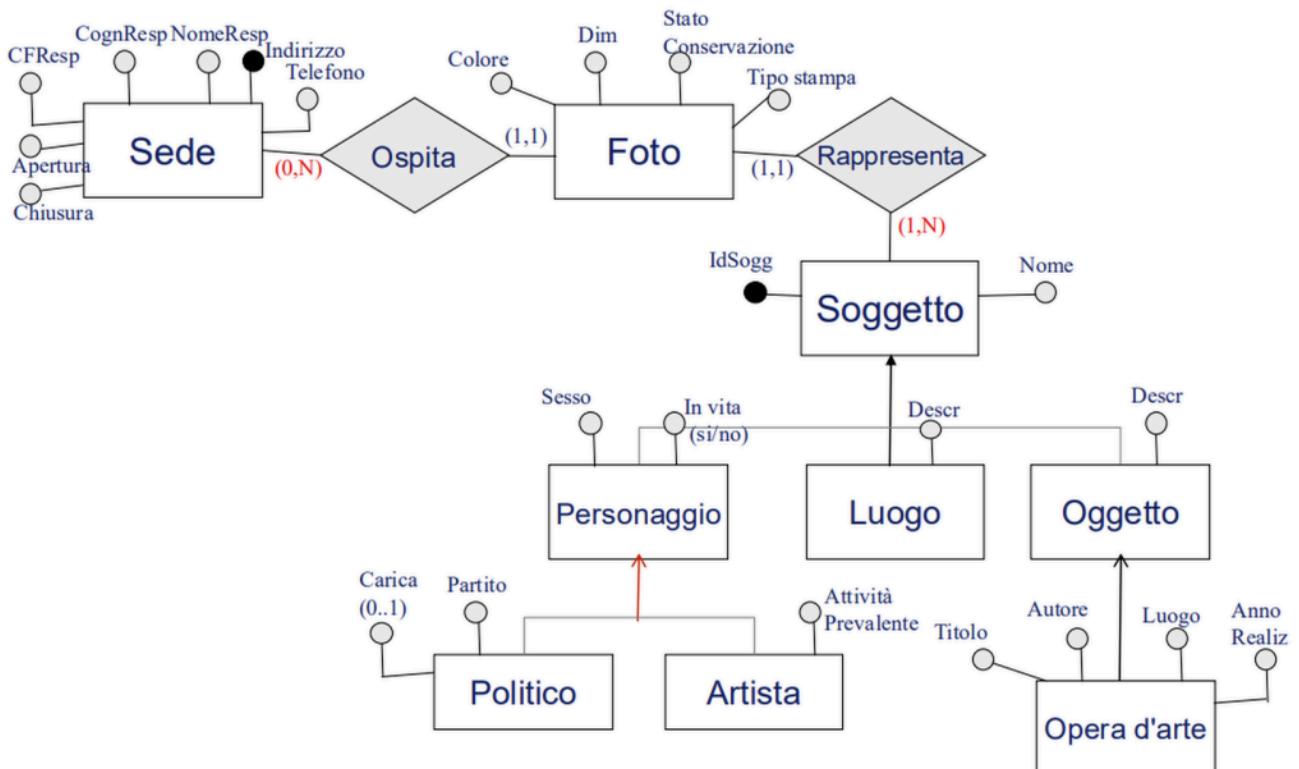
## FRASI RELATIVE AL SOGGETTO “OGGETTO”

Nel caso il soggetto di una foto sia un oggetto, si conoscono: nome dell'oggetto e descrizione. Se l'oggetto è un'opera d'arte, si conoscono anche: il nome dell'opera d'arte, il nome e cognome dell'artista che l'ha realizzata, il luogo dove l'opera è custodita e l'anno di realizzazione.



- Nome opera d'arte ridenominata Titolo
- Nome e Cognome Artista ridenominati Autore

# Integrazione



## Vincoli

Lo schema è completato con documentazione relativa ad eventuali vincoli non espressi nel diagramma ER:

Una persona può essere responsabile di più di una sede contemporaneamente

Le foto sono catalogate rispetto ad un solo soggetto (soggetto dominante)

Per la inclusione oggetto – opera d'arte mantenuti e ridenominati gli **nome** dell'opera d'arte (Titolo) e **nome e cognome dell'artista** (autore) in quanto ritenute informazioni semanticamente differente rispetto al nome dell'oggetto

Valori ammessi

- Colore = {B-N, colori}
- Dim = {picc, med, grande}
- Stato Conservazione = {scarso, sufficiente, buono, ottimo}
- Tipo stampa = {chiaro, opaco}

## Progettazione logica

## Ristrutturazione

Viene sempre consigliato di far sempre prima la ristrutturazione.

In questa avvengono due fasi importanti:

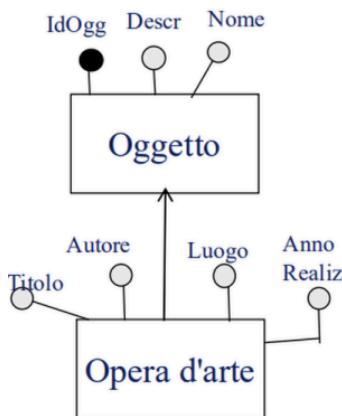
- **Analisi delle ridondanze**
- **Eliminazione delle generalizzazioni**
- **Partizionamento/accorpamento di entità e associazioni**
- **Scelta degli identificatori primari**
- **Diagramma E-R Ristrutturato**

### Analisi delle ridondanze

In base alle ridondanze presenti, eliminarle o modificare e argomentare tale operazione. Se lo schema E-R in generale è fatto bene, questa fase non dovrebbe esistere e dovrebbe soltanto essere argomentata la sua inutilità.

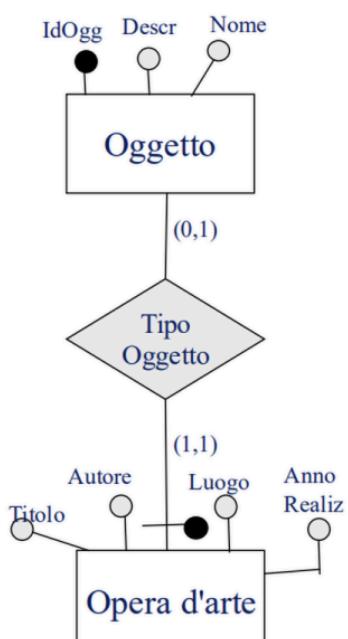
### Eliminazione delle generalizzazioni

Qui si vanno ad eliminare le generalizzazioni dello schema E-R, dando luogo a nuove associazioni (esistono altri metodi, ma questo in generale viene ritenuto il più semplice):

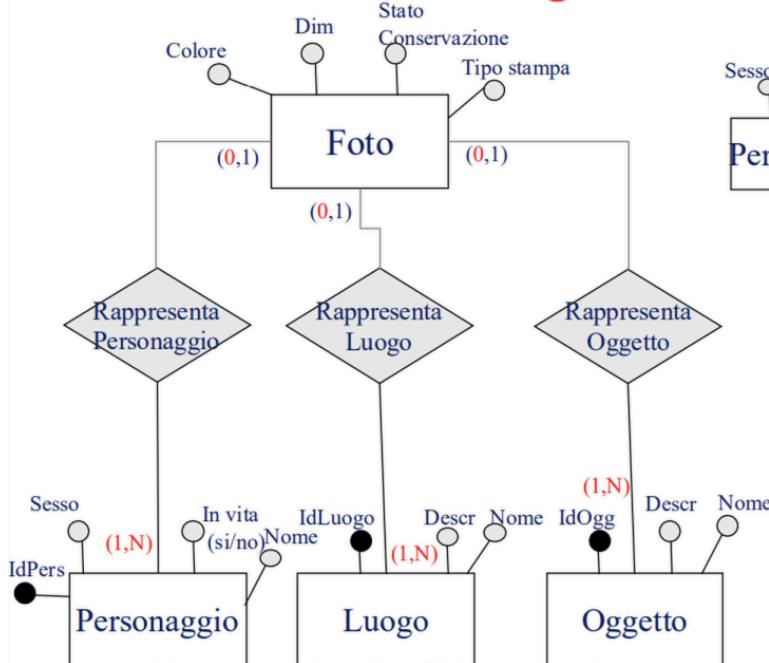


- La generalizzazione è parziale → non si può eliminare l'entità padre
- L'entità figlia ha diversi attributi caratterizzanti che avrebbero valori nulli eliminando l'entità figlia (più della metà degli Oggetti non sono Opere d'arte)

Si sostituisce la generalizzazione con una associazione



## 2. Eliminazione delle generalizzazioni



- Le operazioni fanno riferimento a specifici soggetti (Op. 1, (2), (3))
- Quasi tutte le entità figlie hanno specifici attributi
- La generalizzazione è totale e le istanze sono quasi equamente distribuite tra le entità figlie

Si elimina il padre e si mantengono le figlie

## Partizionamento/accorpamento di entità e associazioni - Scelta degli identificatori primari

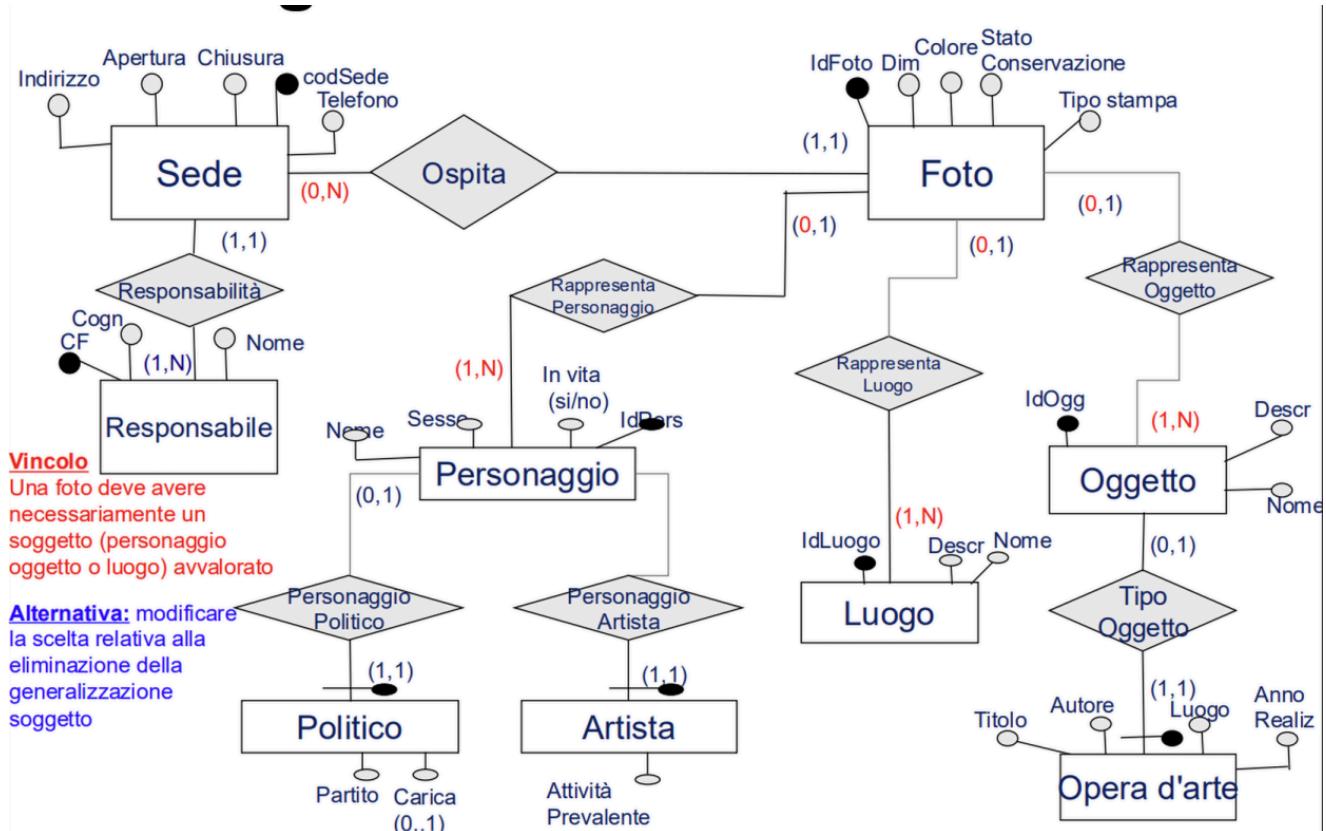
In questa fase andiamo ad accoppare tutti le entità, associazioni e attributi multivalue che possono essere accoperti, nel nostro caso:

- Poiché una persona può essere responsabile di più di una sede si decide di creare una entità per il responsabile
- Non sono presenti attributi multivalue da dover eliminare

Dopodiché, nel caso questo non fosse avvenuto nella fase di progettazione concettuale, si scelgono degli identificatori primari, nel nostro caso:

- Si introduce codSede per Sede al posto di Indirizzo
- Si introduce codFoto per Foto per cui non è stata trovato un identificato in fase di progettazione concettuale
- Tutte le altre entità hanno già un unico identificatore

## Diagramma E-R ristrutturato



## Determinazione carico applicativo

Basandosi sulle [Specifiche delle operazioni](#), andiamo a produrre le tabelle:

## Tavola dei volumi

Concetto	Tipo	Volume
Sede	E	10
Ospita	R	20000
Foto	E	20000
Rappresenta	R	20000
Soggetto	E	6000
Personaggio	E	1500
Luogo	E	2000
Oggetto	E	2500
Politico	E	800
Artista	E	450
Opera d'arte	E	1100

## Tavola delle operazioni

Operazione	Tipo	Frequenza
Op.1	I	50 al giorno
Op.2	I	10 l'anno
Op.3	I	10 al giorno
Op.4	I	2 a settimana
Op.5	I	200 al giorno
Op.6	I	10 al mese

Modalità: I = interattiva

B = batch

## Determinazione costo operazione

(Questo passaggio va effettuato per tutte le operazioni)

### Es. Operazione 1

Concetto	Costrutto	Accessi	Tipo
Soggetto	Entità	2000	L
Rappresenta	Relazione	4000	L
Foto	Entità	4000	L
Ospita	Relazione	4000	L
Sede	Entità	4000	L

Valor medio possibili soggetti  
(1500+2000+2500)/3

In media ogni soggetto è rappresentato in due foto

**Operazione 1:** visualizzare l'insieme delle fotografie per un certo tipo di soggetto e la loro dislocazione fisica (50 volte al gg)

Assunzione: costo L = 1, costo S=2

Costo unitario op. 1 è 18000

Costo op. 1: 900000 (considerato frequenza)

## Traduzione in modello E-R relazionale

Sede(codSede, Indirizzo, Telefono, Apertura, Chiusura, CFResp)

Responsabile(CF, Nome, Cogn)

Foto(IdFoto, Dim, Colore, StatoConservazione, TipoStampa,  
codSede)

RappresentaPersonaggio(IdFoto, IdPers)

RappresentaLuogo(IdFoto, IdLuogo) Luogo(IdLuogo, Nome, Descr)

RappresentaOggetto(IdFoto, IdOgg) Oggetto(IdOgg, Nome, Descr)

Personaggio(IdPers, Nome, Sesso, InVita)

Politico(IdPers, Partito, Carica\*)

Artista(IdPers, AttivitaPrevalente)

OperaDAre(IdOgg, Titolo, Autore, AnnoRealiz, LuogoCustodia)

Specificare vincoli di integrità referenziale esplicitati