Nell'ambito delle strutture fisiche di accesso ai dati, descrivere le strutture sequenziali entry sequenced, ad array e ordinate
Seriale: La sequenza delle tuple è indotta dal loro ordine di inserimento accesso con scan sequenziale. ottimale per le operazioni di scrittura (avviene alla fine del file) e lettura sequenziali. Svantaggi:le cancellazioni lasciano spazio inutilizzato, le modifiche che creano aumento delle dimensioni della tupla non possono essere gestite 'in loco', l'accesso a singoli dati richiede la scansione dell'intero file. **Ad array**: Le tuple sono disposte in un array e la loro posizione dipende dal valore di uno o più campi indice array composto da n blocchi contigui, ognuno di m slot disponibili per le tuple (array di n*m posizioni), ogni tupla ha un indice numerico i ed è posizionata nell'i-esima celletta dell'array (vantaggi: accesso diretto,svantaggi: possibile solo quando le tuple hanno lunghezza fissa, le cancellazioni lasciano celle vuote, si può applicare solo quando la chiave assume valori consecutivi). **Struttura sequenziale ordinata**: La sequenza delle tuple dipende dal valore, in ciascuna tupla, di un campo di ordinamento (campo chiave). Vantaggi:efficienti le operazioni basate sul valore della chiave. Svantaggi:inserimenti e modifiche richiedono la riorganizzazione delle tuple già presenti.

Nell'ambito delle strutture primarie per l'organizzazione dei file, presentare le strutture hash-based descrivendo in modo preciso cosa accade in caso di collisione.
Posizione delle tuple è determinata da un algoritmo di calcolo (hash-based). alloca al file B bucket (spesso adiacenti). la locazione fisica di una tupla dipende da un campo chiave (composto da un numero arbitrario di attributi di una data tabella). il metodo di accesso utilizza un algoritmo di hash che, applicato alla chiave, ritorna un valore fra 0 e B−1. l'implementazione consiste in due parti: folding: trasforma il valore chiave in un intero positivo, distribuito uniformemente su un intervallo. hashing: trasforma l'intero positivo in un numero tra 0 e B−1. Collisioni risolte aggiungendo una catena di overflow partendo da ogni pagina. (aggiunge il costo di scandire la catena). la lunghezza media della catena di overflow: aumenta con l'aumentare del coefficiente di riempimento del file, diminuisce con l'aumentare della dimensione del bucce. Vantaggi: assicura un accesso associativo ed efficiente ai dati. Svantaggi: collisioni, le modifiche alla dimensione del file possono richiedere ristrutturazione, ottimizza solo accessi basati sul campo chiave, inefficiente per operazioni su intervalli

Nell'ambito delle strutture fisiche di accesso, discutere la gestione delle tuple nelle pagine
Ogni pagina sequenziale o hash-based ha (assumendo un blocco per pagina): una parte iniziale (block header) e una parte finale (block trailer) contenenti l'informazione di controllo usata dal file system una parte iniziale (page header) e una parte finale (page trailer) contenenti l'informazione di controllo del metodo di accesso. un dizionario di pagina che contiene i puntatori a ogni dato elementare contenuto nella pagina,una parte utile che contiene i dati; generalmente il dizionario e la parte utile crescono come stack contrapposti, un bit di parità per verificare l'integrità dell'informazione contenuta nella pagina.

Illustrare in modo preciso le differenze tra alberi B ed alberi B+ e descrivere quali vantaggi hanno gli alberi B+ rispetto agli alberi B
Come gli alberi B, ma per ogni nodo: P_{FN} indirizza il sottoalbero con chiavi ≥ di K_{FN}; Pj , 0 < j < F_N indirizza il sotto-albero con chiave K: Kj ≤ K < K_{j+1} quindi le foglie contengono tutti i valori della Chiave, i nodi foglia sono fra loro collegati da una catena sulla base dell'ordine della chiave. **Vantaggi:**Alberi B+ forniscono accesso associativo (basato sul valore di una chiave) senza restrizioni sulla posizione fisica delle tuple. Sono efficienti anche nelle interrogazioni di intervallo, l' utilizzo attento di split e merge consente di mantenere l'albero ben bilanciato.È la struttura utilizzata più frequentemente da DBMS relazionali

Cap2
Nell'ambito della gestione dei lock, dire cosa si intende per lock gerarchico
In molti sistemi reali i lock possono essere chiesti su oggetti a granularita differente: XL (exclusive lock) corrisponde al write lock del protocollo normale, SL (shared lock) corrisponde al read lock del protocollo normale ISL (intentional shared lock) esprime l'intenzione di bloccare in modo condiviso uno dei nodi che discende dal nodo corrente IXL (intentional exclusive lock) esprime l'intenzione di bloccare in modo esclusivo uno dei nodi che discende dal nodo corrente, SIXL (shared intentional-exclusive lock) chiede lock condiviso sul nodo corrente e esprime l'intenzione di bloccare in modo esclusivo uno dei nodi che discende dal nodo corrente i lock si richiedono dalla radice e scendendo lungo l'albero: i lock si rilasciano dal nodo locked e salendo lungo l'albero, una transazione puo chiedere un lock SL o ISL su un nodo, solo se ha un lock ISL o IXL sul genitore, una transazione puo chiedere un lock IXL, XL, o SIXL su un nodo, solo se ha gia un lock SIXL o IXL sul genitore.

Proprietà acide
Atomicità (gestore dell'affidabilità): Una transazione è una unità atomica di lavoro non puo lasciare la base di dati in uno stato intermedio (un guasto o un errore prima del commit causano l'UNDO del lavoro fatto fino a quel punto un guasto o un errore dopo il commit possono richiedere il REDO del lavoro fatto precedentemente, se la sua permanenza sulla base di dati non e garantita), **Consistenza(compilatore del DDL)**: L'esecuzione di una transazione non deve violare i vincoli di integrità definiti sulla base di dati, il controllo sul mantenimento dell'integrita puo essere: immediato, durante la transazione (l'operazione che causa la violazione e rifiutata) o differito, alla fine della transazione (se dei vincoli sono violati, l'intera transazione e rifiutata). **Isolamento (gestore della concorrenza**): L'esecuzione di una transazione deve essere indipendente da quella di tutte le altre transazioni concorrenti. **Durabilità(gestore dell'affidabilità)**: Gli effetti di una transazione che ha eseguito commit non devono essere persi (il sistema deve garantire persistenza dei dati anche in caso di malfunzionamenti e guasti)

Problemi che possono sorgere nell'ambito della concorrenza delle transizioni:

perdita di aggiornamenti: Modifiche di una transazione perse perché sovrascritte da una transazione concorrente. **Lettura sporca**: Una transazione legge il risultato intermedio di un'altra transazione che poi viene abortita (e la cui modifica viene quindi annullata). **letture inconsistenti**: Una transazione legge oggetti che un'altra transazione sta modificando: alcune letture sono prima, altre dopo le modifiche. **Aggiornamenti fantasma**: Una transazione osserva solo parte degli effetti di un'altra transazione (osservando uno stato dei dati che non soddisfa i vincoli di integrità). **Inserimento fantasma**: Una transazione valuta due volte un valore aggregato relativo ad un insieme di elementi di un predicato di selezione (ex:Se tra una valutazione e l'altra viene inserita una nuova tupla, i due risultati possono differire)

Nell'ambito del controllo di concorrenza, fornire la definizione di schedule seriale e schedule serializzabile. Si richiede inoltre di spiegare in cosa consiste l'assunzione di commit-projection
Schedule seriale: per ogni transazione ti nello schedule, tutte le operazioni di ti sono eseguite consecutivamente(n transazioni, n! schedule seriali possibili). Schedule serializzabile: schedule non seriale che produce lo stesso risultato di un qualche schedule seriale delle stesse transazioni(richiede nozioni di equivalenza fra schedule, concetti progressivi: view-equivalenza, conflictequivalenza, two-phase locking, timestamp-based).

Livelli di isolamento SQL
Read uncommitted:Non pone alcun vincolo, non chiede lock per leggere, non rispetta i lock esclusivi di altre transazioni, puo presentare tutte le anomalie delle transazioni concorrenti. **Read committed**: Chiede lock condivisi per effettuare le letture, esclude le letture di stati intermedi, non legge dati non committed, evita letture sporche,non garantisce serializzabilita. **Repeatable read**: Chiede 2PL stretto anche per la lettura, lock a livello di tupla, evita tutte le anomalie ad eccezione dell'inserimento fantasma, **Serializable**: Chiede 2PL stretto anche per la lettura e utilizza lock di predicato, evita tutte le anomalie, default.

Si consideri la tecnica di prevenzione dei deadlock basata sui timestamp.(a) Dire quale transazione viene uccisa nel caso:preemptive,non preemptive.Quale timestamp viene assegnato alla successiva attivazione della transazione abortita? Perch´e?
ti richiede lock su x,tj ha lock su x.non interrompente :se ts(ti) < ts(tj): ti aspetta altrimenti: abort(ti), poi rilancia ti con lo stesso ts(ti). interrompente (wound-wait): se ts(ti) > ts(tj): ti aspetta, altrimenti: abort(tj); poi rilancia tj con lo stesso ts(tj). le transazioni uccise devono ripartire con lo stesso timestamp altrimenti rischierebbero di essere sempre uccise (starvation)

Caratterizzare il problema del deadlock e descrivere brevemente i possibili approcci alla sua soluzione (sia per prevenirli sia per risolverli).
Deadlock distribuiti: Situazioni circolari di attesa tra 2 o più nodi. Soluzioni: - timeout: le transazioni rimangono in attesa solo per un tempo massimo prefissato (timeout), allo scadere del tempo se la transazione e ancora in attesa: viene ritornata risposta negativa o la transazione abortita. Rilevazione di deadlock e risoluzione: Rileva la presenza di cicli nel grafo di attesa, non pone vincoli al comportamento del sistema ma controlla le tabelle di lock per rilevare deadlock. Prevenzione: Evita che si verifichi un deadlock utilizzando tecniche preventive che assicurano non vi sara mai mutua attesa. Tecniche preventive di mutua attesa:2PL conservativo: allocazione dei lock all'inizio della transazione, può non essere sempre possibile.

Cap 3
Nell'ambito delle basi di dati distribuite, dire cosa si intende per frammentazione orizzontale e frammentazione verticale. Si richiede inoltre di descrivere le proprietà di correttezza che una relazione frammentata deve soddisfare.
Frammentazione orizzontale: Ogni R_i ha come tuple un sottoinsieme delle tuple di R, ogni R_i può essere interpretato come risultato di una selezione su R. – *Frammentazione verticale*: Ogni R_i ha come schema un sottoinsieme degli attributi di R, ogni R_i può essere interpretato come risultato di una proiezione su R. –*Proprietà di correttezza*: - *Completezza*: Ogni dato di R deve essere presente in qualche suo frammento R_i. – *Ricostruibilità*: R deve essere interamente ricostruibile a partire dai suoi frammenti. Generalmente i frammenti orizzontali sono disgiunti(non hanno tuple in comune), mentre i frammenti verticali includono la chiave primaria di R(garantisce ricontruibilità)

Nell'ambito delle basi di dati distribuite, dire cosa si intende per frammentazione ed allocazione. Descrivere inoltre i livelli di trasparenza che derivano dalla distinzione tra frammentazione ed allocazione
Frammentazione e allocazione dei dati la framm consente di organizzare i dati stessi in modo tale da garantire una loro distribuzione efficiente e ben organizzata. La framm può essere orizzontale. Lo schema di allocazione descrive il mapping delle relazioni o dei frammenti ai server che li memorizzano consentendo il passaggio da una descrizione logica ad una descrizione fisica dei dati. Ogni frammento corrisponde a un file a livello fisico ed è allocato su uno specifico server. L'allocazione può essere non-ridondante quando ogni frammento o relazione è allocato su un solo server o ridondante quando almeno un frammento o relazione è allocato su più di un server. **Livelli di trasparenza**: Distinzione fra frammentazione e allocazione consente di scrivere applicazioni a diversi livelli: dal piu astratto e indipendente dalla frammentazione dei dati, al piu concreto dipendente anche dalla loro allocazione fisica. **frammentazione:** il programmatore non ha bisogno di conoscere la frammentazione, non ha bisogno di conoscere l'allocazione. **allocazione:** il programmatore deve conoscere la struttura dei frammenti, non ha bisogno di conoscere l'allocazione. **linguaggio**: il programmatore deve conoscere la struttura dei frammenti deve conoscere l'allocazione. **assenza di trasparenza**: ogni DBMS accetta il suo proprio 'dialetto' SQL: il sistema e eterogeneo e i DBMS non supportano uno standard comune per l'interoperabilità.

Classificazione delle transazioni

Richieste remote:transazioni read-only composte di un numero arbitrario di query SQL, tutte le query sono indirizzate a un singolo DBMS remoto, il DBMS remoto puo essere solo interrogato. **Transazioni remote**: composte da un numero qualsiasi di comandi SQL (select, insert, delete, update), tutti i comandi sono indirizzati a un singolo DBMS remoto, ogni transazione scrive su un singolo DBMS. **Transazioni distribuite**: composte da un numero qualsiasi di comandi SQL (select, insert, delete, update) indirizzati a un numero arbitrario di DBMS remoti, ogni comando SQL si riferisce a un singolo DBMS, ogni transazione puo aggiornare piu di un DBMS. **Richieste distribuite**: transazioni arbitrarie composte da un numero qualsiasi di comandi SQL (select, insert, delete, update) indirizzati a un numero arbitrario di DBMS remoti,ogni comando si puo riferire a qualsiasi DBMS, richiede un ottimizzatore di query distribuite.

Nell'ambito delle basi di dati distribuite, dire cosa si intende per serializzabilità globale di transazioni distribuite e dire quali proprietà sono soddisfatte
Serializzabilità globale: La serializzabilità locale non garantisce serializzabilità globale. La serializzabilità globale richiede l'esistenza di uno schedule seriale S equivalente a tutti gli schedule locali Si risultanti a ogni nodo. Proprietà: - Schedule globale conflict-serializzabile,la serializzabilità globale è garantita se ogni scheduler usa 2PL stretto ed esegue il commit in modo atomico quando tutte le sotto-transazioni ai vari nodi hanno tutte le risorse. – Schedule globale seriale, la serializzabilità globale è garantita se ogni transazione distribuita acquisisce un singolo timestamp e lo usa in tutte le richieste a tutti gli scheduler che fanno controllo della concorrenza basato sui timestamp(richiede assegnamento di TS globale).

Deadlock distribuiti
Rilevazione di deadlock distribuiti: Attivata periodicamente ai vari DBMS del sistema, ogni DBMS integra nuove sequenze di attesa con le condizioni di attesa locali, analizza il grafo risultante per rilevare il deadlock e comunica le sequenze di attesa ad altri DBMS. Protocolli per il commit distribuito: Permettono a una transazione di raggiungere la decisione corretta di commit o abort su tutti i nodi che partecipano alla transazione. Il più diffuso è il two-phase-commit.

Illustrare il funzionamento del protocollo two phase commit in assenza di guasti
La decisione commit/abort presa dalle parti è registrata da una terza parte, che ratifica la decisione. Distingue i server che partecipano alla decisione(gestori di risorse RM) e il processo coordinatore(gestore della transazione TM). Funziona tramite scambio rapido di messaggi tra TM e RM e scritture dei record nei log. TM e RM mantengono log per assicurare resistenza ai guasti. Quando un RM si dichiara ready per una transazione perde la propria autonomia e deve rimanere soggetto alla decisione del TM. Prima della dichiarazione della decisione o se si è dichiarato non-ready, RM può abortire autonomamente facendo UNDO dei suoi effetti, senza partecipare al protocollo two-phase-commit. Si compone di 2 fasi: - Prima fase: TM scrive il record di prepare nel suo log e manda un messaggio prepare a tutti gli RM e imposta il timeout. Ogni RM, se in stato affidabile, scrive nel log il record ready e trasmette al TM il messaggio ready, che indica la scelta di partecipare al protocollo; se è in stato non affidabile, manda un messaggio non ready e termina la propria partecipazione al protocollo. TM raccoglie i messaggi di risposta e scrive il log: global commit se tutti gli RM hanno risposto ready, global abort, se almeno un RM ha risposto non ready o il timeout è scattato e non tutti i messaggi sono stati ricevuti.- Seconda fase: TM trasmette la sua decisione globale a tutti gli RM e imposta il timeout. Ogni RM in stato ready scrive nel log il record relativo alla decisione globale e manda un ACK al TM. La decisione globale viene decisa in loco. TM raccoglie tutti gli ACK dagli RM coinvolti nella seconda fase. Se il timeout scade, stabilisce un altro timeout e ripete il messaggio a tutti gli RM dai quali non ha ricevuto ACK. Quando tutti gli ACK sono arrivati scrive il record complete nel suo log

Cap4
Nell'ambito dei dati semistrutturati e XML, descrivere il significato dei componenti delle espressioni FLOWR
Sono simili al costrutto SELECT-FROM-WHERE di SQL ma anziché essere applicate a tabelle associano variabili e valori e utilizzano tali associazioni per produrre risultati. Sono: - FOR, per dichiarare variabili che permettono di iterare sugli elementi di un documento; - LET, per dichiarare variabili associate al risultato dell'espressione, eventualmente associandole a quelle introdotte con for; - WHERE, per esprimere condizioni e filtra le tuple prodotte dal for e dalla let; - ORDER, per ordinare le tuple prodotte dal for e dalle let; - RETURN, per generare il risultato.

Nell'ambito dei dati semistrutturati, dire cosa rappresenta il document type definition (DTD). Si richiede inoltredi scrivere un semplice esempio di DTD
DTD e XML schema: Permettono di definire un modello per i documenti ovvero dettano il tipo dei documenti cioè i tag ammessi e le loro priorità e le regole di annodamento.Elementi (DTD): contenenti altri elementi figli, <!ELEMENT PRODOTTO (DESCRIZIONE)>; con PCDATA <!ELEMENT DESCRIZIONE (#PCDATA)>; contenuto misto, <!ELEMENT ARTICOLO (#PCDATA|PRODOTTO)>; elementi vuoti , <!ELEMENT ARTICOLO EMPTY>

Nell'ambito dell'XML Schema: (a) Indicare quale `e la differenza fra elementi semplici ed elementi complessi. (b) Elencare e descrivere gli indicatori di ordinamento per gli elementi
Elementi semplici (XML Schema): Gli elementi semplici non possono contenere altri elementi o attributi, possono essere solo tipi standard o tipi di dati derivati da questi, possono avere valori di default, possono avere valori fissi e possono avere associate delle restrizioni. Elementi complessi (XML Schema): Gli elementi complessi possono contenere attributi o altri elementi e possono utilizzare degli indicatori.b) Indicatori di ordinamento: - Any, qualunque elemento, in qualunque ordine; - All, tutti gli elementi , in qualunque ordine; -Choice, uno e un solo elemento; - Sequence, tutti gli elementi, nell'ordine specificato. Indicatori della cardinalità: - max0ccurs, max numero di occorrenze; - min0ccurs, min numero di occorrenze. Incatori di raggruppamento: -Group name; - Group reference.

Dire quali sono le condizioni che rendono un documento XML ben formato
Un documento è detto ben formato se:inizia con il prologo; tutti gli elementi hanno tag iniziali e finali;La nidificazione dei tag è corretta;gli attributi sono correttamente codificati;i valori sono correttamente codificati

Cap 5
Nell'ambito delle basi di dati attive, indicare tutte le proprietà.
E' necessario avere garanzie che l'interferenza tra le diverse regole e l'attivazione a catena non generi anomalie, del tipo di: -*terminazione*: per qualunque stato iniziale e qualunque sequenza di modifica, le regole producono uno stato finale, cioé non devono esserci cicli infiniti di attivazione (é questa l'anomalia più importante). -*confluenza*: per qualunque stato inziale e qualunque sequenza di modifiche, le regole producono uno stato finale (terminazione) e inoltre producono un unico stato finale, indipendente dall'ordine in cui i trigger vengono eseguiti (é significativa quando il sistema presenta del non determinismo nella scelta delle regole da eseguire). -*determinismo delle osservazioni*: per qualunque stato iniziale e qualunque sequenza di modifiche, le regole terminano e producono un unico stato finale, indipendente dall'ordine in cui le regole vengono eseguite (confluenza), e producono la stessa sequenza di azioni visibili.

Nell'ambito delle basi di dati attive, descrivere i componenti dei trigger insieme con i livelli di granularità e le modalità di esecuzione
DBMS diversi possono differire nella gestione dell'attivazione dei trigger. La creazione dei trigger fa parte del Data Definition Language. Le sue componenti sono: ***evento*** (primitive SQL per la manipolazione dei dati, ad es.insert, delete, update), ***condizione*** (predicato booleano espresso in SQL) e *azione* (sequenza di primitive SQL generiche o procedura). I trigger presentano **2 livelli di granularità**: -*tupla* (row-level), in cui l'attivazione viene avviene per ogni tupla coinvolta nel'evento; -*primitiva* (statement-level), in cui l'attivazione avviene una sola volta per ogni evento e si applica a tutte le tuple coinvolte nell'evento. I passi per l'**esecuzione dei trigger** in SQL sono i seguenti: si eseguono prima i trigger statement e poi, in ordine, quelli before now, i test di integrità sulla tabella, i trigger after row, e infine quelli after statement. Se vi sono più trigger della stessa categoria, l'ordine di esecuzione viene scelto dal sistema in un modo che dipende dall'implementazione.

Cap6
Nell'ambito delle basi di dati ad oggetti, dire cosa si intende per identità e uguaglianza tra due oggetti
Identità e uguaglianza:Tra gli oggetti sono definite le seguenti relazioni:-Identità: Richiede che gli oggetti abbiano lo stesso identificatore. uguaglianza superficiale:Richiede che gli oggetti abbiano lo stesso stato,cioè lo stesso valore per proprietà omologhe.-Uguaglianza profonda: Richiede che le proprietà che si ottengono seguendo i riferimenti abbiano gli stessi valori, non richiede l'uguaglianza dello stato.

Nell'ambito delle basi di dati ad oggetti, dire cosa sono le gerarchie di generalizzazione. Fornire anche un esempio di gerarchia
Gerarchie di generalizzazione (o di ereditarietà):Tra i tipi (e le classi) di una base di dati a oggetti è possibile definire una gerarchia di ereditarietà.Una sotto-classe eredita lo stato e il comportamento della sopra-classe.Tutti gli oggetti delle sotto-classi appartengono automaticamente alle sopra-classi. Il ruolo della gerarchia di ereditarietà è lo stesso dei linguaggi di programmazione orientati agli oggetti. Esiste tuttavia una importante differenza: gli oggetti di un programma sono oggetti di breve durata, gli oggetti
di una base di dati sono oggetti di lunga durata, con conseguenze non banali

Nell'ambito delle basi di dati ad oggetti, dire cosa si intende per definizione co-variante e definizione controvariante. Si richiede di fornire un esempio di definizione co-variante e di descrivere che problemi pu`o introdurre
Se si ridefinisce un metodo di una sottoclasse, i suoi parametri possono essere definiti in 2 modi: -*co-variante*: i parametri sono sotto-tipi dei parametri della sopraclasse; -*contro-variante*: i parametri sono sopra-tipi dei parametri della sopraclasse. La soluzione co-variante è più diffusa, ma crea dei problemi nei parametri di ingresso.Ex: Non è possibile garantire a compile-time che l'invocazione del metodo su File sia corretta (il file può essere un Sorgente, ma il parametro Owner passato non è un Programmatore)

Cap 7
Nell'ambito della basi di dati per il supporto alle decisioni, descrivere i concetti fatto, misura e dimensione del modello multidimensionale. Si richiede inoltre di fornire un esempio per ognuno dei concetti fornendo una spiegazione
La rappresentazione dei dati ad alto livello prescinde dai criteri di memorizzazione e favorisce l'analisi. I concetti rilevanti di questo modello, sono: il *fatto*, che é il concetto sul quale centrare l'analisi; la *misura*, che é una proprietà atomica del concetto da anlizzare; e la *dimensione*, che descrive una prospettiva lungo la quale effettuare l'analisi. Esempio: in una catena di negozi, la vendita é il fatto, l'incasso é la misura, e il tempo e il luogo sono le dimensioni

Nell'ambito delle data warehouse, descrivere lo schema a *stella* e lo schema a *fiocco di neve* evidenziando le differenze e i vantaggi/svantaggi di uno schema rispetto all'altro
Schema a stella: E' costituito da 3 componenti principali: -*Tabella dei fatti*: una tabella principale che memorizza i dati del data mart; la sua chiave è composta da attributi che sono riferimenti alle chiavi delle tabelle dimensione. –*Tabelle dimensione*: sono relazioni ausiliare che memorizzano i dati relativi alle dimensioni dell'analisi; hanno una chiave semplice (composta da un solo attributo) e gli attributi non chiave rappresentano i livelli della dimensione o qualche loro proprietà, e sono tipicamente testuali/descrittivi. *Vincoli di integrità referenziale*, ognuno dei quali collega un attributo della tabella dei fatti a una tabella dimensione. Schema a fiocco di neve (snow flake): Risulta da una

normalizzazione (anche parziale) di uno schema a stella, e permette di evitare ridondanze eccessive nelle dimensioni. E' una normalizzazione da fare con attenzione, in quanto generalmente porta degrado nelle prestazioni, dalla tabella dei fatti si raggiungono tutte le tabelle delle dimensioni.

Nell'ambito del data mining, dire cosa si intende per confidenza e supporto di una regola di associazione
Vi sono delle regole di classificazione, che classificano nuovi fenomeni in classi predefinite e presentano fondamentalmente 2 proprietà: ***supporto***, che rappresenta la probabilità che siano presenti in una transazione entrambi gli elementi di una regola, e ***confidenza***, che costituisce invece la probabilità che sia presente in una transazione la conseguenza di una regola, essendo presente la premessa

Elencare e descrivere le principali componenti dell'architettura di un data warehouse
Base di dati per il supporto alle decisioni (OLAP) con le seguenti caratteristiche: -*Integrata*: riunisce i dati di diverse sorgenti informative preesistenti e rappresenta i dati in modo univoco, riconciliando le eterogeneità delle diverse rappresentazioni (nomi, struttura, codifica). -*Dati in forma aggregata*: i dati delle sorgenti sono aggregati sulla base di opportune coordinate (es.: tempo, collocazione geografica) ed é orientata ai dati (non alle appllicazioni), cioé le basi di dati operazionali sono costruite a supporto dei singoli processi operativi o applicazioni, e la data warehouse é costruita attorno alle principali entità del patrimonio informativo aziendale. -*Dati storici/temporali*: tiene l'evoluzione storica delle informazioni con un ampio orizzonte temporale e indicazione di elementi di tempo. -*Fuori linea e non volatile*: tipicamente formata in modo asincrono e periodico rispetto alle sorgenti. -*Autonoma*: fisicamente separata dalle sorgenti informative, per vari motivi fra cui: ragioni tecniche; non esiste un'unica base di dati operazionale che contiene tutti i dati di interesse; la data warehouse deve essere integrata; i dati di interesse sarebbero comunque diversi (in quanto devono essere mantenuti dati storici ed aggregati); l'analisi dei dati richiede organizzazioni speciali dei dati e metodi di accesso specifici; degrado generale delle prestazioni senza la separazione.

Nell'ambito delle basi di dati e World Wide Web, dire cosa `e il common gateway interface (CGI) e in che modo possono essere inviati dei parametri.
E' il primo e il più semplice standard architetturale proposto per risolvere il problema della creazione dinamica delle pagine. Si basa su un semplice concetto, quello di utilizzare l'URL della richiesta HTTP per invocare un programma presente sul server, che calcolerà la pagina da restituire al client.

Nell'ambito delle basi di dati per il supporto alle decisioni, descrivere in modo completo le operazioni di roll-up e drill-down. Si richiede di fornire anche un esempio: Operazioni su dati multidimensionali: -Slice-and-dice (seleziona e proietta): seleziona un sottoinsieme delle celle di un cubo. -Roll-up (aggrega i dati): applica una funzione aggregativa (tipicamente somma) sui dati aggregati di un cubo; é possibile effettuarla su misure additive (su cui ha senso fare la somma lungo una dimensione). -Drill-down (disaggrega i dati): é l'operazione inversa del roll-up e aggiunge dettaglio ad un cubo disaggregandolo lungo una o più dimensioni.

Es Log
Con undo: -U(..Ox..) /Ox:=Bx
- D(..Ox..) /insert Ox:=Bx
- I(..Ox..) /delete Ox
Con redo: - U(..Ox..) /Ox=Ax
- D(..Ox..) /delete Ox
- I (..Ox..) /insert Ox=Ax

read(x, ts):sempre accettata,leggi la versione xk tale che WTM(xk) = max{WTM(xi) | WTM(xi) ≤ ts } RTM(xk):= max(RTM(xk),ts)
write(x, ts): trova la versione xj tale che WTM(xj) = max{WTM(xi) | WTM(xi) ≤ ts }, ts < RTM(xj) richiesta rifiutata, transazione abortita, altrimenti richiesta accettata; crea nuova versione xk RTM(xk):=ts; WTM(xk):=ts

read(x, ts): ts < WTM(x) rifiutata transazione abortita,altrimenti:accettata; RTM(x) := max(RTM(x),ts)
write(x, ts): ts < WTM(x) o ts < RTM(x) rifiutata, transazione abortita.altrimenti accettata; WTM(x) := ts