GLI ATTRIBUTI DI UNA TAB PRIVATA POSSONO ESSERE:

- IDENTIFICATORI: identifica univocamente un rispondente

- QUASI IDENTIFICATORI
  ATTRIBUTI CHE POSSONO ESSERE COLLEGATI CON DATI ESTERNI

- CONFIDENZIALI
  ATTRIBUTI CON INFO CONFIDENZIALI (MALATTIA)

- NON CONFIDENZIALI (COLORE PREFERITO)

K-ANONIMITY

OBIETTIVO: PROTEGGERE IDENTITÀ RISPONDENTI QUANDO RILASCIO MICRODATI

PROBLEMA: NON BASTA TOGLIERE LE INFO SENSIBILI (NOME, TEL...) PERCHÉ POSSONO ESSERE USATE ALTRE INFO ESTERNE PER IDENTIFICARE I RISPONDENTI (RACE, ZIP, SEX)

K-ANON: OGNI TUPLA NELLA TABELLA NON PUÒ ESSERE CORRELATA A MENO DI K RISPONDENTI.

K-ANON + QUASI-IDENTIFICATORI:

OGNI RELEASE DI DATI DEVE ESSERE FATTA IN MODO CHE OGNI COMBINAZIONE DI QUASI ID È IDENTIFICATA CON ALMENO K RISPONDENTI.

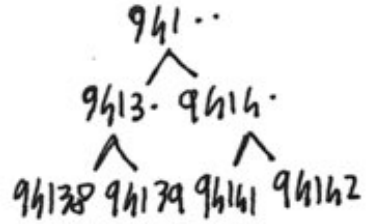
POICHÉ SAREBBE IMPOSSIBILE SAPERE I DATI PUBBLICI A DISPOSIZIONE, K-ANON CERCA DI RISOLVERE IL PROBLEMA ALLA RADICE FACENDO IN MODO CHE I RISPONDENTI SIANO INDISTINGUIBILI RISPETTO A UN SET DI ATTRIBUTI

↓ PER FARE QUESTO

OGNI QUASI ID DEVE AVERE ALMENO K OCCORRENZE

TECNICHE PROTEZIONE MICRODATA (SAMPLING SWAPPING NOISE) RENDONO DATI MENO VERITIERI.
SE VOGLIAMO MANTENERLI IL PIÙ VERITIERI POSSIBILI USIAMO K-ANON + GENERALIZZAZIONE/SOPPRESSIONE.

GENERALIZZAZIONE:
SOSTITUIRE IL VALORE CON UN VALORE PIÙ GENERALE.
ZIP PUÒ ESSERE GENERALIZZATO TOGLIENDO AD OGNI STEP LA CIFRA MENO SIGNIFICATIVA.
ABBIAMO UNA GERARCHIA DI DOMINI RAPPRESENTABILE COME UN ALBERO:



SOPPRESSIONE:
SOPPRIMO TUPLE SENSIBILI.
QUESTO METODO PUÒ RIDURRE IL NUMERO DI GENERALIZZAZIONI NECESSARIE PER SODDISFARE K-ANON.

UNA TABELLA T_j È GENERALIZZAZIONE DI T_i

- $|t_j| \leq |T_i|$
- IL DOMINIO DI OGNI ATTRIBUTO DI T_j È UGUALE O GENERALIZ DEL DOMINIO DELL'ATTRIBUTO DI T_i
- TUPLA $t_j \xrightarrow[\text{INIETTIVA}]{\text{FUNZIONE}}$ TUPLA t_i

  IL VALORE DI OGNI ATTRIBUTO IN t_j È UGUALE O GENERALIZ DEL VALORE DELL'ATTRIBUTO CORRISPONDENTE IN t_i

OBIETTIVO È CREARE UNA TAB CHE MANTENGA PIÙ INFO POSSIBILI RISPETTANDO K-ANON.

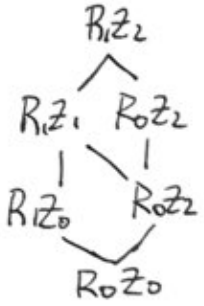
~~GENERALIZZAZIONE~~

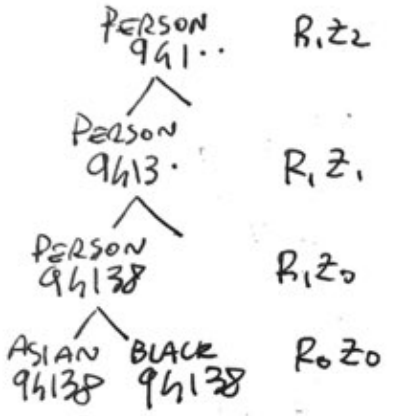
GENERALIZZAZIONE K-MINIMALE CON SOPPRESSIONE SI BASA SUL CONCETTO DI DISTANCE VECTOR:

LUNGHEZZA DEL CAMMINO UNICO TRA DOMINIO $T_i \in T_j$.

POSSIAMO DEFINIRE UN ORDINE PARZIALE TRA I DISTANCE VECTOR.

SI PUO' COSTRUIRE UNA GERARCHIA DI DISTANCE VECTOR (RETICOLO)





GENERALIZZAZIONE K-MINIMALE CON SOPPRESSIONE

- $T_i \preceq T_j$
- MAXSUP

T_j E' GENERALIZ K-MINIMALE DI T_i SE

1. T_j ASSICURA K-ANON FACENDO SOPPRESSIONE MINIMALE
2. $|T_i| - |T_j| \leq$ MAXSUP
   NON SOPPRIME PIU' DI QUANTO SIA PERMESSO
3. NON C'E' UN'ALTRA SOLUZIONE CON DISTANCE VECTOR PIU' CORTO DI T_j

UNA TABELLA PRIVATA PUO' AVERE PIU' DI UNA POSSIBILE GENERALIZ K-MINIMALE. CRITERI:

- MINIMUM ABSOLUTE DISTANCE
- MINIMUM RELATIVE DISTANCE
- MAXIMUM DISTRIBUTION
  NUMERO PIU' GRANDE DI TUPLE DISTINTE
- MINIMUM SUPPRESSION
  SOPPRIME MENO TUPLE POSSIBILI = MAGGIORE CARDINALITA'

GENERALIZ E SOPPRES POSSONO ESSERE FATTE A LIVELLI DI GRANULARITA' DIVERSI: (2)

SOPPR → GENER ↓	tuple	attribute	cell
attribute	AG-TS	AG-AS ∈ AG	AG-CS
cell	CG-TS	CG-AS	CG-CS ∈ CG

ALGORITMI AG-TS E AG- ATTRIBUTO TUPLA

- SAMARATI ALGO (RICERCA BINARIA)

ALGORITMO USA SIA GENERALIZ CHE SOPPR DELLE TUPLE CON ATTRIBUTI QUASI ID E TROVA SOL K-MINIMALE USANDO CRITERIO MINIMUM ABSOLUTE DISTANCE.

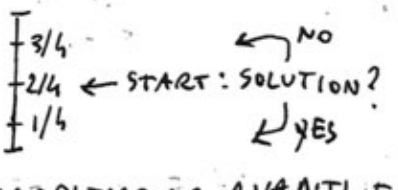
DATA UNA GERARCHIA DI DOMINI CI SONO DIFFERENTI CAMMINI CHE SI POSSONO SCEGLIERE.

OGNI CAMMINO HA UN MINIMO LOCALE CHE RAPPRESENTA LA TAB CHE SODDISFA K-ANON E MANTIENE PIU' INFO POSSIBILI.

IL MINIMO LOCALE E' IL NODO PIU' BASSO (LOWEST).

COME TROVARE GENERALIZ K-MINIMALE:

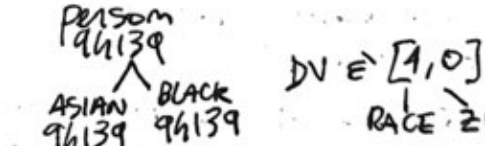
PERCORRERE TUTTI I CAMMINI SAREBBE TROPPO ONEROSO QUINDI FACCIO RICERCA BINARIA PERCHE' SE NON C'E' SOLUZIONE AD ALTEZZA h ALLORA NON C'E' NEI NODI PIU' BASSI DI h.



ALGORITMO VA AVANTI FINCHE' NON RAGGIUNGE LA PIU' BASSA h CON UN DV CHE SODDISFA K-ANON.

QUESTO APPROCCIO RICHIEDE LA COMPUTAZIONE DI TUTTE LE TAB GENERALIZ. PER EVITARLO INTRODUCIAMO CONCETTO: DISTANCE VECTOR BETWEEN TUPLES.

DV TRA DUE TABELLE X E Y E' IL VETTORE COMPOSTO DELLA DISTANZA TRA GLI ELEMENTI DI X E Y E I LORO ANTENATI COMUNI (= GENERALIZZAZIONE).



COSTRUIAMO COSI' UNA TABELLA

	t_1	t_2	t_3	...
t_1	[0,0]			...
t_3	[0,1]			
...				

→ TUTTE LE TUPLE

↓ OUTLIERS

• K-OPTIMIZE

PARTIZIONAMENTO DEL DOMINIO DEGLI ATTRIBUTI IN INTERVALLI "ORDINATI": I VALORI DI UN INTERVALLO I CHE PRECEDE UN ALTRO INTERVALLO J PRECEDONO I VALORI DELL'INTERVALLO J.

ZIP {94138, 94139} {94141, 94142}
I J

! CI DEVE ESSERE ORDINE TRA I QI

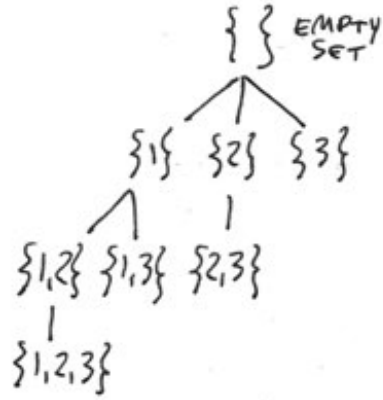
{[ASIAN][BLACK][WHITE]}{[94138][94139]}
1 2 3 4 (5) INDEX

C'È ORDINE TOTALE NELL'INTERVALLO.

LA GENERALIZZAZIONE È L'UNIONE DEGLI INDICI INDIVIDUALI:

[94138][94139] → [94138 or 94139]

K-OPTIMIZE CREA UN SETENUMERATION TREE:



ALBERO GARANTISCE ESISTENZA DI UN CAMMINO TRA LA ROOT E OGNI NODO.

ALGORITMO:
AD OGNI NODO M VIENE CALCOLATO IL COSTO DELLA GENERALIZ IN QUEL NODO E QUESTO COSTO VIENE COMPARATO AL COSTO MIGLIORE FINO A QUEL MOMENTO.
SE IL NUOVO COSTO È PIU' BASSO DEL MIGLIORE ALLORA IL NUOVO COSTO DIVENTA IL NUOVO COSTO MIGLIORE.

K-OPTIMIZE PRUNA UN NODO (E IL SUO SOTTOALBERO) QUANDO DETERMINA CHE NESSUNO DEI SUOI DISCENDENTI PUO' ESSERE OTTIMALE CIOE' QUANDO IL COSTO DELLA GENERALIZ DEL NODO E' PIU' ALTO DEL COSTO MIGLIORE FINO A QUEL MOMENTO.

VENGONO ANCHE CANCELLATI QUEI NODI CHE CONTENGONO GLI ELEMENTI CHE FACEVANO PARTE DEL NODO CANCELLATO - ANCHE SE NON FANNO PARTE DEL SOTTOALBERO.
(SE CANCELLO {1,3} CANCELLO ANCHE {1,2,3})

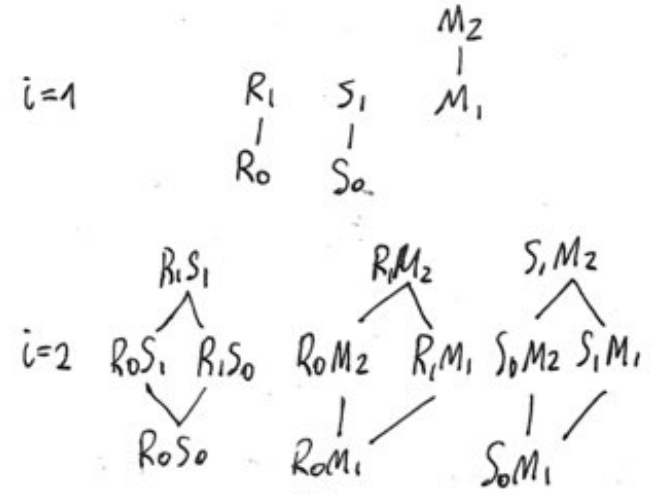
• INCOGNITO

OBIETTIVO: COMPUTARE GENERALIZ K-MINIMALE

IDEA: LA K-ANON RISPETTO A UN SUBSET DI QI È COND NECESSARIA (MA NON SUFFICIENTE) PER LA K-ANON DI TUTTO IL QI.
→ INCOGNITO ESCLUDE A PRIORI ALCUNE GENERALIZZAZIONI

ALGORITMO:
BOTTOM-UP BREADTH-FIRST SEARCH NEL DOMINIO
- CONTROLLA K-ANON PER OGNI ATTRIBUTO IN QI E SCARTA QUELLI CHE NON SODDISFANO K-ANON
- METTE I RIMANENTI IN COPPIE E CONTROLLA K-ANON
- QUINDI IN TRIPLETTE...
- E COSI' VIA FINCHE' TUTTO IL SET E' STATO CONSIDERATO

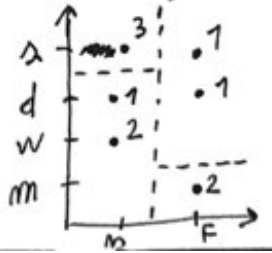
APPROCCIO BOTTOM-UP
AD OGNI COMBINAZIONE:
QUANDO UNA GENERALIZ SODDISFA K-ANON ALLORA ANCHE TUTTE LE DIRETTE GENERALIZ



ALGORITMI _CS E CG_ CELLA

• MONDRIAN

UNA FUNZIONE MULTIDIMENSIONALE GENERALIZ DEFINISCE UN SET DI REGIONI MULTIDIM.
LE REGIONI CORRISPONDONO AGLI INTERVALLI DEFINITI NEL SINGOLO SCENARIO.
OGNI TUPLA RAPPRESENTA UN PUNTO NELLO SPAZIO MULTIDIMENSIONALE E LE COORD SONO I VALORI DELL'ATTRIBUTO QI.



! SERVE ORDINE TOTALE

{a} {d OR w} {a OR d} {m}

K-ANON PROTEGGE DA IDENTITY DISCLOSURE MA NON DA ATTRIBUTE DISCLOSURE.

DUE POSSIBILI ATTACCHI:

- ATTACCO OMOGENEITA'
  SE UN ATTACCANTE CONOSCE IL VALORE DEI QI DI UN RISPONDENTE E CONOSCE CHE QUESTO RISPONDENTE FA PARTE DELLA POPOLAZIONE NELLA TAB ALLORA L'ATTACCANTE PUO' INFERIRE QUAL È IL VALORE DELL'ATTRIBUTO SENSIBILE PER IL RISPONDENTE NOTO POICHÉ TUTTE LE TUPLE CON QUEL QI HANNO LO STESSO VALORE.

- ATTACCO BACKGROUND KNOWLEDGE
  QUANDO L'ATTACCANTE PUO' RIDURRE L'INCERTEZZA DI UN VALORE DI UN ATTRIBUTO SENSIBILE GRAZIE A INFORMAZIONI ESTERNE CHE LUI POSSIEDE.
  QUESTO ATTACCO DIVENTA PIU' DIFFICILE ALL'AUMENTARE DI l PERCHÉ ALL'ATTACCANTE SERVE PIU' CONOSCENZA.

→ l-DIVERSITY
UN BLOCCO q E' l-DIVERSO SE CONTIENE ALMENO l DIFFERENTI VALORI. TAB T E' l-DIVERSA SE TUTTI I BLOCCHI SONO l-DIVERSI.

ATTACCHI BASATI SULLA DISTRIBUZIONE DEI VALORI ALL'INTERNO DEL BLOCCO q:

- SKEWNESS ATTACK
  I RISPONDENTI NEL BLOCCO HANNO UNA DISTRIBUZIONE MOLTO DIVERSA RISPETTO ALLA POPOLAZIONE O RISPETTO ALLA TAB PUBBLICA RILASCIATA.

- SIMILARITY ATTACK
  QUANDO I VALORI NEL BLOCCO SONO SEMANTICAMENTE SIMILI.

→ t-CLOSENESS
LA DISTRIBUZIONE DEI VALORI SENSIBILI NELLA TAB RILASCIATA DEVE ESSERE SIMILE ALLA DISTRIBUZIONE DELLA TAB PRIVATA.

RILASCI MULTIPLI

K-ANON ASSUME CHE I DATI IN UNA TAB PUBBLICATA NON VENGANO MODIFICATI. MA UNA TAB PUO' ESSERE SOGGETTA A NUMEROSE MODIFICHE E QUINDI A DIVERSE PUBBLICAZIONI NEL TEMPO..
UN ATTACCANTE PUO' USARE LE DIFFERENZE NELLE TAB PER INFERIRE QUALCOSA (ATTACCO INTERSEZIONE)

→ m-INVARIANCE
UNA SEQ DI RILASCI RISPETTA m-INVARIANCE SE
- CANCELLO/TENGO ALMENO m TUPLE TRA I DUE RILASCI
- I DATI SENSIBILI DEVONO APPARIRE PIU' DI UNA VOLTA IN OGNI CLASSE DI EQ
- PER OGNI TUPLA t LE CLASSI DI EQ DELLA TUPLA t SONO CARATTERIZZATE DALLO STESSO SET DI DATI SENSIBILI

UN ATTACCANTE NON RIESCE AD ASSOCIARE MENO DI m DIFFERENTI DATI SENSIBILI.

DIFFERENTIAL PRIVACY

DATI SEMANTICI: LE TECNICHE PER PROTEGGERE I DATI SEMANTICI HANNO L'OBIETTIVO DI PROTEGGERE LA PRIVACY DEI RISPONDENTI CHE NON COMPAIONO NELLA TAB RILASCIATA.

DIFFERENTIAL PRIVACY GARANTISCE CHE IL RILASCIO DI UNA TABELLA DI MICRODATI NON DIVULGHI INFO SENSIBILI RIGUARDO NESSUN INDIVIDUO CHE E' O NON E' RAPPRESENTATO DA UNA TUPLA NELLA TAB.

OBIETTIVO E' RILASCIARE UN DATASET CHE PERMETTA DI ESTRARRE LE PROPRIETA' RIGUARDO LA POPOLAZIONE PROTEGGENDO LA PRIVACY DEI SINGOLI INDIVIDUI.

QUINDI LA PROB CHE UN ATTACCANTE INFERISCA QUALCOSA RIGUARDO UN RISPONDENTE NON E' LEGATA ALLA PRESENZA O MENO DELLA TUPLA DEL RISPONDENTE NELLA TAB RILASCIATA.

MACRODATA / MICRO DATA

~~MACRODATA~~ SI RILASCIANO MICRODATI INVECE DI MACRODATI COSÌ DA AUMENTARE LA FLESSIBILITÀ E LA DISPONIBILITÀ DI INFORMAZIONI.

- IDENTITY DISCLOSURE
  QUANDO UNA COMBINAZIONE DI ATTRIBUTI IDENTIFICANTI PUÒ PORTARE ALL'IDENTIFICAZIONE DI UN INDIVIDUO.

- ATTRIBUTE DISCLOSURE
  QUANDO USANDO UNA COMBINAZIONE DI ATTRIBUTI INDIRETTAMENTE IDENTIFICANTI UN VALORE DI UN ATTRIBUTO PUÒ ESSERE ASSOCIATO A UN INDIVIDUO.

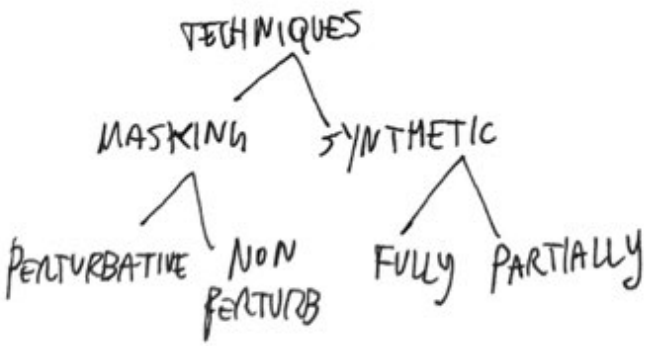
- INFERENTIAL DISCLOSURE
  QUANDO POSSO INFERIRE CON ALTA PROB ALCUNE INFO DALLE PROPRIETÀ STATISTICHE DEI DATI PUBBLICATI.

MICRODATA

SONO LE INFORMAZIONI DI UN RISPONDENTE (NAME, RACE, DOB, SEX, ZIP).
ATTRIBUTI CLASSIFICATI IN
- IDENTIFICATORI
- QUASI ID
- CONFIDENZIALI
- NON CONF

I DATI POSSONO ESSERE
- CONTINUI : OPERAZIONI ARITMETICHE E NUMERICHE SUI DATI.
- CATEGORICI : INSIEME DI DATI SPECIFICI SUI QUALI NON SI POSSONO FARE OP ARITMETICHE.

TECNICHE DI PROTEZIONE
- EVITARE REIDENTIFICAZ DEI RISPONDENTI
- MANTENERE LE PROPRIETÀ STAT DEI DATI



MASKING

I DATI ORIGINALI SONO TRASFORMATI PER PRODURRE NUOVI DATI CHE MANTENGONO LE PROPRIETÀ STATISTICHE E LA CONFIDENZIALITÀ DEI RISPONDENTI.

- NON PERTURBATIVE
  I DATI ORIGINALI NON SONO MODIFICATI MA ALCUNI SONO CANCELLATI

  + SAMPLING TAB PUBBLICA CONTIENE SOLO UN SAMPLE DELLA TAB PRIVATA.

  + LOCAL SUPPRESSION SOPPRIME QUELLE CELLE CHE ESPORREBBERO I RISPONDENTI.

  + RECODING DOMINIO DI UN ATTRIBUTO E' PARTIZIONATO IN INTERVALLI. NELLA TAB PUBBLICA PUBBLICO ETICHETTE.

  + TOP/BOTTOM CODING
    I VALORI PIÙ ALTI DEL TOP CODE E PIÙ BASSI DEL BOTTOM CODE VENGONO SOSTITUITI DAL TOP/BOTTOM CODE.

  + GENERALIZZAZIONE SOSTITUISCO ~~RIDUCO IL DOMINIO~~ I VALORI CON VALORI PIÙ GENERALI. GENERALIZ BASATA SULLA GERARCHIA DEI DOMINI.

- PERTURBATIVE
  I DATI ORIGINALI SONO MODIFICATI

  + RESAMPLING SOSTITUISCO I VALORI CON UNA MEDIA CALCOLATA SU UN SAMPLE PRESO DALLA TAB PRIVATA.

  + COMPRESSION LA TAB VIENE INTERPRETATA COME UNA IMMAGINE E VIENE APPLICATO UN ALG LOSSY.

  + ROUNDING SOSTITUISCO I VALORI CON DEI VALORI ARROTONDATI (CI SONO DEI ROUNDING SET).

  + RANDOM NOISE PERTURBA UN ATTRIBUTO SOMMANDO O MOLTIPL PER UN VALORE RANDOM.

  + SWAPPING MODIFICO UN SUBSET SCAMBIANDO I VALORI DI UN SET CONTENENTE DATI SENSIBILI CON I VALORI DI ALTRE TUPLE.

MACRODATA

DATI AGGREGATI: STATISTICHE DI UNA POPOLAZIONE

TECNICHE DI PROTEZIONE

- TABELLE FREQUENZA (FREQUENCY)

  UNA CELLA CONTIENE NUMERO DI RISPONDENTI CHE HANNO CERTA CARATTERISTICA.

  + SAMPLING — CALCOLO LE STATISTICHE SU UN CAMPIONE DI RISPONDENTI RAPPRESENTATIVO.

  + REGOLE SPECIALI — DEFINISCO UN LIVELLO DI DETTAGLIO SULLE INFO CHE VOGLIO RILASCIARE.

  + REGOLE SOGLIA — UNA CELLA E' SENSIBILE SE IL VALORE E' SOTTO UNA CERTA SOGLIA.

- TABELLE GRANDEZZA (MAGNITUDE)

  OGNI CELLA CONTIENE IL VALORE AGGREGATO DI UNA QUANTITA' DI INTERESSE.

  + P-PERCENT RULE

    PERSONE DEL CAMPIONE CHE CONOSCONO I PROPRI VALORI SI METTONO INSIEME CERCANDO DI CALCOLARE I RESTANTI VALORI. LA CELLA E' ESPOSTA SE SI RIESCE A CALCOLARE IL VALORE IN MODO TROPPO ACCURATO.

    CELLA PROTETTA SE $\sum_{i=c+2}^{N} x_i \geq \frac{p}{100} x_1$

    ($\sum_{i=c+2}^{N}$: COALIZIONE UTENTI; x_i: VALORI; $\frac{p}{100} x_1$: VALORE DA PROTEGGERE)

  + PQ-RULE

    $$\frac{q}{100} \sum_{i=c+2}^{N} x_i \geq \frac{p}{100} x_1$$

    q: ABILITA' DI STIMARE GLI ALTRI VALORI

  + (m,k)-RULE

    CELLA E' SENSIBILE SE M O MENO RISPONDENTI CONTRIBUISCONO AD ALMENO IL K % DEL TOTALE.

→ COSA FACCIO QUANDO CELLA E' SENSIBILE

- RISTRUTTURO LA TAB COLLASSANDO RIGHE E COLONNE
- SOPPRESSIONE PRIMARIA (CANCELLO CELLA) O SECONDARIA (CHIUDO CANALI INFERENZA)

USER PRIVACY PREFERENCES

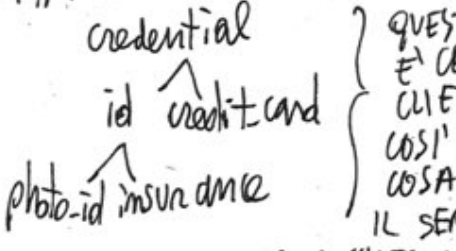
LE RISORSE POSSONO ESSERE ACCEDUTE DA OVUNQUE. PER REGOLARE L'ACCESSO I SERVER CHIEDONO AGLI UTENTI DI RILASCIARE INFO TRAMITE CERTIFICATI DIGITALI. E' IMPORTANTE DEFINIRE MECCANISMI CHE PERMETTANO AL CLIENT E AL SERVER DI SPECIFICARE LE PREFERENZE PRIVACY.

LA SOLUZIONE CHE PERMETTE A UN SERVER DI REGOLARE L'ACCESSO AI SERVIZI CHE OFFRE (SENZA CHE L'UTENTE ABBIA DEGLI ACCOUNT) SI CHIAMA ABAC (ATTRIBUTE BASED ACCESS CONTROL): DEFINISCE LE CONDIZIONI CHE IL CLIENT DEVE SODDISFARE PER POTER ACCEDERE AL CONTENUTO.

QUANDO UN SERVER RICEVE UNA RICHIESTA INVIA AL CLIENT LE CONDIZIONI DA SODDISFARE PER POTER ACCEDERE AL SERVIZIO.
IL CLIENT RILASCIA UN CERTIFICATO DIGITALE (A.E. LE CREDENZIALI) FIRMATO DA UNA TERZA PARTE CHE CERTIFICA CHE IL POSSESSORE DEL CERTIFICATO POSSIEDA GLI ATTRIBUTI SCRITTI NEL CERTIFICATO.

LE INFO CHE UN CLIENT PUO' RILASCIARE SONO CONTENUTE IN UN PORTFOLIO.
OGNI CREDENZIALE E' CARATTERIZZATA DA:
ID, EMITTENTE, ATTRIBUTI, TIPO.

IL TIPO DETERMINA L'INSIEME DEGLI ATTRIBUTI.
I TIPI SONO ORGANIZZATI IN UNA GERARCHIA:



QUESTA GERARCHIA E' CONDIVISA TRA CLIENT E SERVER COSI' IL SERVER SA COSA CHIEDERE.
IL SERVER COMPONE LA RICHIESTA IN BASE ALLA GERARCHIA. LA POLITICA E' UNA FORMULA BOOLEANA TIPO cond(term1, ...) CHE SUPPORTA OPERAZIONI COME >, <=.

LE CREDENZIALI POSSONO ESSERE
- ATOMICHE: POSSONO SOLO ESSERE RILASCIATE TUTTE INSIEME
- NON ATOMICHE: IL CLIENT PUO' SELETTIVAMENTE RILASCIARE UN SOTTOINSIEME DEGLI ATTRIBUTI CERTIFICATI DALLE CREDENZIALI.

GLI ATTRIBUTI IN UNA CREDENZIALE HANNO: TIPO, NOME, VALORE.

CLIENT E SERVER DEVONO COSTRUIRE UN RAPPORTO DI FIDUCIA. QUESTO RAPPORTO E' COSTRUITO STEP-BY-STEP ATTRAVERSO LO SCAMBIO DI CREDENZIALI. 5

IL RILASCIO DI INFO SENSIBILI E' REGOLATO DALLE ACCESS CONTROL POLICIES.

ANCHE IL SERVER DEVE RILASCIARE DELLE CREDENZIALI (a.e. OSPEDALE X).
PER AVERE ACCESSO A UNA RISORSA CLIENT E SERVER DEVONO CREARE UNA STRATEGIA (UNA SEQUENZA DI SCAMBI DI CERTIFICATI) PER SODDISFARE LE ACCESS CONTROL POLICIES DI ENTRAMBI:

CLIENT PRIVACY PREFERENCES

IL CLIENT PUO' ESPRIMERE LE PROPRIE PREF RIGUARDO LE CREDENZIALI DA RILASCIARE: (PUO' CAPITARE CHE PIU' CREDENZIALI VADANO BENE)

- FINE GRAINED SPECS
  RIFLETTE LA SENSIBILITA' CHE IL CLIENT HA RIGUARDO LE PROPRIE CREDENZIALI.
- INHERITANCE (eredità)
  IL MODELLO APPROFITTA DELLA GERARCHIA: LE PREF PRIVACY VENGONO EREDITATE.
- PARTIAL ORDER
  IL DOMINIO DELLE PREFERENZE PRIVACY DOVREBBE ESSERE CARATTERIZZATO DA UN ORDINE PARZIALE CHE PERMETTE DI DETERMINARE SE UN PEZZO DI INFO PERSONALE E' PIU' O MENO SENSIBILE RISPETTO A UN ALTRO.
- SENSITIVE ASSOCIATION
  IL RILASCIO DI UNA COMBINAZIONE DI ATTRIBUTI POTREBBE ESSERE PIU' O MENO SENSIBILE DEL RILASCIO DELLE SINGOLE COMPONENTI.
- DISCLOSURE CONSTRAINT
  CLIENT SPECIFICA LE RESTRIZIONI RIGUARDO LE COMBINAZIONI DI ATTRIBUTI CHE SI VOGLIONO RILASCIARE.
- CONTEXT BASED
  LE PREFERENZE PRIVACY POTREBBERO VARIARE IN BASE AL CONTESTO.

- HISTORY BASED
  LA DECISIONE DI RILASCIARE UNA CREDENZIALE PIUTTOSTO CHE UN'ALTRA DIPENDE DA QUALI CREDENZIALI SONO STATE RILASCIATE IN PASSATO.
- PROOF OF POSSESSION
  CLIENT PUO' RILASCIARE LA PROVA DI AVERE UN CERTIFICATO O LA PROVA DI SODDISFARE LE CONDIZIONI RICHIESTE.

SERVER PRIVACY PREFERENCES

IL SERVER REGOLA GLI ACCESSI DEL CLIENT TRAMITE ABAC. LA POLITICA DI ACCESSO POTREBBE ESSERE SENSIBILE E IL SERVER NON VUOLE DIVULGARLA. DUE POLICY:

- DISCLOSURE POLICY
  IL SERVER DEFINISCE CON GRANULARITA' COME DIVULGARE LA POLITICA DI RILASCIO DELLE RISORSE.
- POLICY COMMUNICATION
  LA COMUNICAZIONE DELLA POLITICA DEL SERVER DEVE PROTEGGERE LA PRIVACY DEL SERVER E ALLO STESSO TEMPO DEVE ESSERE CHIARA PER IL CLIENT.

① COST SENSITIVE TRUST NEGOTIATION

OGNI CREDENZIALE NEL CLIENT PORTFOLIO E' ASSOCIATA A UN COSTO CHE MISURA IL VALORE DELL'EVENTUALE RILASCIO.
IL CLIENT VUOLE RILASCIARE QUELLE CREDENZIALI CHE HANNO MINORE COSTO.

OBIETTIVO: MINIMIZZARE IL COSTO DELLE CREDENZIALI SCAMBIATE DURANTE UNA NEGOZIAZIONE.
MA CALCOLARE LA SOL CON COSTO MINIMO E' NP-HARD. DUE SOLUZIONI:

- POLICY GRAPH
  IL PESO DI UN VERTICE RAPPRESENTA IL COSTO DI UNA CREDENZIALE.
  TROVARE LA SOL MINIMA E' COME TROVARE IL DAG MINIMO.
  (SI USA UNA VARIANTE DI DJKSTRA)
- GREEDY STRATEGY

② POINT BASED TRUST NEGOTIATION

IL SERVER ASSOCIA UN NUMERO DI PUNTI AD OGNI TIPO DI CREDENZIALE. QUESTO VALORE RAPPRESENTA L'ATTENDIBILITA' PERCEPITA DAL SERVER RIGUARDO L'EMITTENTE DELLE CREDENZIALI.
IL SERVER SETTA UNA SOGLIA PER ACCEDERE AL SERVIZIO. PER ACCEDERE IL CLIENT DEVE RILASCIARE UN INSIEME DI CREDENZIALI LA CUI SOMMA DEVE ESSERE SUPERIORE ALLA SOGLIA.

ANCHE IL CLIENT DA' UN PUNTEGGIO PRIVACY ALLE PROPRIE CREDENZIALI. PIU' E' ALTO IL PUNTEGGIO MENO IL CLIENT E' DISPOSTO A RILASCIARE LA RISORSA.
OBIETTIVO DEL CLIENT: RAGGIUNGERE SOGLIA MINIMIZZANDO IL PIU' POSSIBILE IL PUNTEGGIO PRIVACY.

IL SERVER NON RENDE PUBBLICA LA PROPRIA POLICY E IL CLIENT IL PROPRIO PUNTEGGIO PRIVACY.

③ LOGICAL BASED MINIMAL CREDENTIAL DISCLOSURE

POTREBBE ESSERE DIFFICILE PER UN UTENTE ESPRIMERE NUMERICAMENTE LE PROPRIE PREFERENCE PRIVACY.

VIENE SCELTO UN METODO QUALITATIVO.
ASSUNZIONI: - CREDENZIALI SONO SINGLETON
- SERVER POLICY E' PUBBLICA

OBIETTIVO: TROVARE LA NEGOZIAZIONE CHE MEGLIO SODDISFA LE PREFERENZE UTENTE.
CLIENT PUO' SCEGLIERE MANUALMENTE LA MIGLIORE MA QUANDO LE CREDENZIALI AUMENTANO LE POSSIBILITA' SONO TROPPE.

CREIAMO UNA TABELLA DI NEGOZIAZIONE

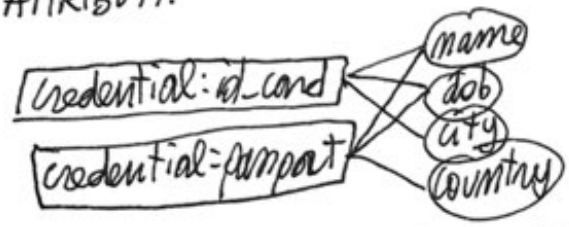
subset	credenziali	
	0 1 0 ...	1: rilasciata
	1 0 1 ...	0: non rilasciata

PER CONFRONTARE I SUBSETS LE PREFERENZE SONO CREATE TRAMITE LA PARETO COMPOSITION: UN SUBSET DOMINA UN ALTRO SE E' PIU' PICCOLO.

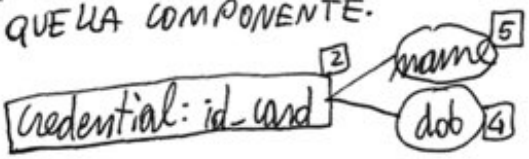
④ PRIVACY PREFERENCES IN CREDENTIAL BASED INTERACTIONS

IL CLIENT PORTFOLIO VIENE MODELLATO COSÌ CHE IL CLIENT POSSA SPECIFICARE LE SUE PREFERENZE RIGUARDO LA PRIVACY IN MANIERA GRANULARE (COMPRESI I CONSTRAINTS SULLE SINGOLE COMPONENTI)

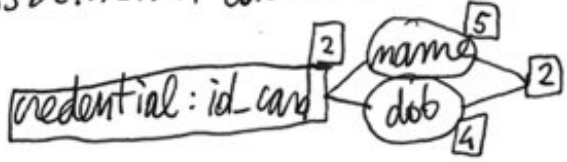
IL CLIENT PORTFOLIO È MODELLATO COME SE FOSSE UN GRAFO BIPARTITO CON UN VERTICE PER OGNI CREDENZIALE E PER OGNI ATTRIBUTO E GLI ARCHI CHE COLLEGANO CREDENZIALI E ATTRIBUTI.



IL CLIENT ESPRIME LE PREFERENZE TRAMITE DELLE ETICHETTE CHE RAPPRESENTANO QUANTO IL CLIENT DÀ VALORE AL RILASCIO DI QUELLA COMPONENTE.



CI POTREBBERO ESSERE SITUAZIONI IN CUI IL RILASCIO COMBINATO DI ALCUNE COMPONENTI HA UN COSTO MAGGIORE O MINORE RISPETTO ALLA SOMMA DELLE ETICHETTE DELLE COMPONENTI. PER QUESTO È POSSIBILE SPECIFICARE DELLE ASSOCIAZIONI CON ETICHETTE:



- SENSITIVE VIEW
  L'ASSOCIAZIONE RILASCIA PIÙ INFO RISPETTO ALLA COMPOSIZIONE DELLE ETICHETTE DELLE COMPONENTI
  (a.e. dob + city)

- DEPENDENCIES
  L'ASSOCIAZIONE RILASCIA MENO INFO RISPETTO ALLA COMPOSIZIONE DELLE ETICHETTE DELLE COMPONENTI
  (a.e. city + country)

CI SONO QUE TIPI DI CONSTRAINTS CHE NON POSSONO ESSERE ESPRESSI CON LE ETICHETTE: 6

- FORBIDDEN VIEW
  UN SUBSET DI COMPONENTI IL CUI RILASCIO COMBINATO È PROIBITO.

- DISCLOSURE LIMITATIONS
  AL MASSIMO n ELEMENTI NEL SET POSSONO ESSERE DIVULGATI INSIEME.

NON TUTTI I GRUPPI (SUBSETS) DI CREDENZIALI E ATTRIBUTI POSSONO ESSERE COMUNICATI AL SERVER NELLA RICHIESTA DI ACCESSO AL SERVIZIO RICHIESTO.

UN GRUPPO D DI COMPONENTI RAPPRESENTA UNA DISCLOSURE SOLO SE SODDISFA 3 CONDIZIONI:

1. CERTIFIABILITY
   OGNI ATTRIBUTO È CERTIFICATO DA ALMENO UNA CREDENZIALE
2. ATOMICITY
   SE UN ATTRIBUTO CERTIFICATO DA UNA CREDENZIALE ATOMICA È DIVULGATO (DISCLOSED) ALLORA TUTTI GLI ATTRIBUTI NELLA CREDENZIALE SONO DIVULGATI
3. ASSOCIATION EXPOSURE
   SE TUTTI GLI ATTRIBUTI O LE CREDENZIALI CHE FORMANO UNA ASSOCIAZIONE SONO DIVULGATI ALLORA ANCHE L'ASSOCIAZIONE È DIVULGATA.

LA SENSIBILITÀ DEL RILASCIO È CALCOLATA COME LA SOMMA DI TUTTE LE SENSIBILITÀ (ATTRIBUTI + CREDENZIALI + ASSOCIAZIONI) CHE STO RILASCIANDO.

OBIETTIVO: TROVARE UN SUBSET CHE SODDISFI LA RICHIESTA DEL SERVER AVENDO MINORE SENSIBILITÀ POSSIBILE.

↓ COME TROVARE MINIMAL DISCLOSURE

? . TROVARE UNA DISCLOSURE CHE SODDISFI ALMENO UNA DELLE DISGIUNZIONI
. CLIENT PRESENTA UNA CREDENZIALE DEL TIPO RICHIESTO CHE CERTIFICA LA PROPRIETÀ

. PROBLEMA NP-HARD
. USO GRAFO PER FARE EURISTICHE O USO SAT SOLVER

ENCRYPTION AND FRAGMENTATION

IN AMBITO CLOUD IL PROPRIETARIO DEI DATI PERDE IL CONTROLLO DEI PROPRI DATI PERCHÉ LI SALVA ESTERNAMENTE, LASCIANDOLI POTENZIALMENTE ESPOSTI.
I DATI DEVONO ESSERE PROTETTI DAL CLOUD PROVIDER (honest-but-curious) PERCHÉ E' AUTORIZZATO A SALVARE, GESTIRE E PROCESSARE I DATI MA NON A LEGGERNE IL CONTENUTO.
~~COME FACCIAMO A GARANTIRE LA CONFIDENZ DEI DATI:~~
~~- ENCRYPTION DEI DATI PRIMA DI MEMORIZ.~~
~~- FRAGMENTATION: QUANDO L'INFO SENSIBILE E'L'ASSOCIAZIONE TRA I DATI POSSO SALVARE DIFFERENTI PEZZI DI DATI IN FRAMMENTI NON LINKABILI.~~

PROTECTION REQUIREMENTS

QUELLO CHE E' SENSIBILE DEVE ESSERE CONFIDENZ.

- SENSITIVE ATTRIBUTES
  ALCUNI ATTRIBUTI SONO SENSIBILI E I LORO VALORI DOVREBBERO ESSERE CONFIDENZIALI (SSN, CREDIT CARD, ...)
- SENSITIVE ASSOCIATIONS
  ALCUNE VOLTE E' PIU' SENSIBILE L'ASSOCIAZIONE RISPETTO AI SINGOLI ATTRIBUTI (PAZIENTE + MALATTIA)

CONFIDENTIALITY CONSTRAINTS:
GRUPPI DI ATTRIBUTI LA CUI UNIONE NON DOVREBBE ESSERE VISIBILE.

ESEMPIO
Att = {SSN, Name, Race, Job, Disease, Treatment, Ins}
$C_1 = \{SSN\}$
$C_2 = \{Name, Disease\}$
$C_3 = \{Name, Ins\}$
$C_4 = \{Disease, Ins\}$
$C_5 = \{Race, Job, Ins\}$

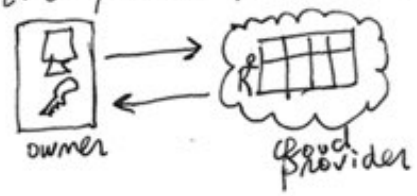
TECNICHE DI PROTEZIONE:
- ENCRYPTION
  CRIPTO I DATI PRIMA DI MEMORIZZARLI ESTERNAMENTE COSÌ DA RENDERLI LEGGIBILI SOLO AGLI UTENTI CON LA CHIAVE.
  LA CRIPTAZIONE VIENE FATTA CON GRANULARITA': TABELLA, COLONNA, TUPLA, CELLA.
  - TABELLA: TUTTA LA TAB DEVE ESSERE INVIATA AL CLIENT PER ESSERE DECRIPTATA
  - COLONNA: PROVIDER PUO' FARE SOLO PROIEZIONE SE NON VUOLE DECRIPTARE TUTTO
  - CELLA: TROPPE OP DI CRIPTAZ/DECRIPTAZ
  - TUPLA E' IL MIGLIOR TRADE-OFF

LE QUERY VENGONO FATTE O DIRETTAMENTE SUL CRIPTATO O TRAMITE METADATI SUL CRIPTATO (INDICI).

- FRAGMENTATION
  QUANDO L'INFO SENSIBILE E' L'ASSOCIAZIONE TRA I DATI POSSO SALVARE DIFFERENTI PEZZI DI DATI IN FRAMMENTI (VISTE VERTICALI) COSÌ DA NON VIOLARE I CONFIDENTIALITY CONSTRAINT.

DATA PROTECTION PARADIGMS

① ENCRYPTION AND INDEXING



I DATI SONO CRIPTATI A LIVELLO DI TUPLA PRIMA DI ESSERE ESTERNALIZZATI, SONO INOLTRE ASSOCIATI CON DEGLI INDICI CHE VENGONO USATI DAL CLOUD PROVIDER PER ESEGUIRE QUERY.

② ENCRYPTION MODEL
Re TABELLA CRIPTATA DOVE
- tid TUPLE ID RANDOM
- enc TUPLA CRIPTATA
- ELENCO DI INDICI $I_i \dots I_j$

tid	enc	I_R	I_j	I_d	I_i
1	~	α	δ	I	ρ
2	~	α	δ	h	ρ
3	~	β	ε	θ	σ

DIRECT HASH FLAT BUCKET PARTITION

IN BASE A COME LA FUNZIONE INDEX MAPPA IL PLAINTEXT NEL CRIPTATO CLASSIFICHIAMO LE TECNICHE DI INDEXING:
- DIRECT INDEX
  MAPPA OGNI VALORE A UN VALORE DI INDEX DIVERSO.

direct over Race

α
α
β
γ
δ
β

- BUCKET INDEX
MAPPA DIFFERENTI VALORI ALLO STESSO VALORE DELL'INDICE MA OGNI VALORE E' MAPPATO A UN SOLO VALORE DELL'INDICE.

· PARTITION BASED
DIVIDE IL DOMINIO IN GRUPPETTI DI VALORI CONTIGUI E ASSOCIA UNA ETICHETTA AD OGNUNO

due intervalli
[100, 150]
[151, 200]

(OPERAZIONI)

· HASH BASED
ADOTTA UNA FUNZIONE DI HASH h CHE GENERA COLLISIONI.
IL VALORE DI INDICE CHE RAPPRESENTA t[A] E' GENERATO CON h(t[A])

h(teacher) =
h(farmer) =
h(secretary) = δ

- FLATTENED INDEX
MAPPA OGNI VALORE A UN GRUPPO DI INDICI IN MODO CHE TUTTI GLI INDICI ABBIANO LO STESSO NUMERO DI OCCORRENZE (FLATTENING). OGNI INDICE RAPPRESENTA UN SOLO VALORE.

ⓐ QUERY EVALUATION
METTERE I DATI SUL CLOUD DEVE ESSERE TRASPARENTE PER L'UTENTE FINALE QUINDI UTENTE DEVE POTER FARE UNA QUERY SULLO SCHEMA ORIGINALE.
QUESTA QUERY DEVE ESSERE TRADOTTA IN UNA QUERY CHE LAVORA SUL CRIPTATO.
LA TRADUZIONE DIPENDE DAL TIPO DI INDICE.

ESEMPIO

q: SELECT Att
FROM R
WHERE Cond

q_p: SELECT tid, enc
FROM R^e
WHERE $Cond_p^e$ AND $Cond_{p'}^e$

q_v: SELECT Att
FROM Decrypt(Rp.enc, k)
WHERE $Cond_v$ AND $Cond_{p'}$

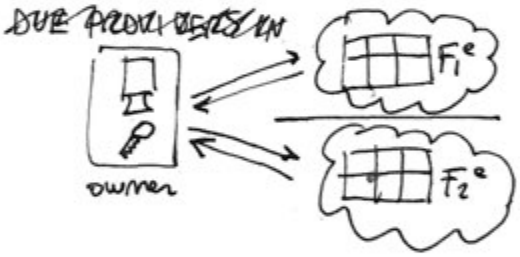
q: SELECT Name
FROM Patients
WHERE Race = White
Job = Teacher
Treat = Panac

q_p: SELECT tid, enc
FROM Patients^e
WHERE I_n = α AND I_j = β

q_v: SELECT Name
FROM Decrypt(Rp.enc, k)
WHERE Job = Teacher
treat = panac

---

② TWO CAN KEEP A SECRET
(SERVER NON COMUNICANTI)



DUE PROVIDERS INDIPENDENTI MEMORIZZANO UNA PARTE DEI DATI. I DUE FRAMMENTI HANNO UNA CHIAVE IN COMUNE COSI' DA POTERLI RIUNIRE. LE ASSOCIAZIONI SENSIBILI SONO PROTETTE PARTIZIONANDO GLI ATTRIBUTI TRA I FRAMMENTI. GLI ATTRIBUTI SENSIBILI SONO SEMPRE CRIPTATI.

ⓐ FRAGMENTATION MODEL
I CONFIDENTIALITY CONSTRAINTS SONO GARANTITI DALLA COMBINAZIONE DI FRAMMENTAZIONE VERTICALE E CRIPTAZIONE E DALL'ASSUNTO CHE I PROVIDERS NON SI PARLINO.
UNA RELAZIONE VIENE DIVISA TRA DUE FRAMMENTI: GLI ATTRIBUTI SENSIBILI SONO CRIPTATI E LE ASSOCIAZIONI SENSIBILI SONO PROTETTE DIVIDENDO GLI ATTRIBUTI TRA I DUE PROVIDERS.
UN FRAMMENTO E' CORRETTO SE RISPETTA TUTTI I CC DEFINITI DALL' OWNER.

UN FRAMMENTO DEVE MEMORIZZARE TUTTI GLI ATTRIBUTI DELLA RELAZIONE ORIGINALE COSI' DA GARANTIRE CHE IL CONTENUTO DELLA RELAZIONE POSSA ESSERE RICOSTRUITO A PARTIRE DA $F = \{F_1, F_2, E\}$ ↳ GRUPPO ATTRIBUTI CRIPTATI

ESEMPIO

$F_1 = \{Name, Race, Job\}$

$F_2 = \{Disease, Treatment\}$

$E = \{SSN, Ins\}$

A LIVELLO FISICO F_1^e E F_2^e MEMORIZZANO GLI ATTRIBUTI IN F_i IN CHIARO E QUELLI IN E IN MODO CRIPTATO.
I DUE FRAMMENTI DEVONO AVERE UN ATTRIBUTO IN COMUNE COSI' DA RICOSTRUIRNE IL CONTENUTO.

ESEMPIO

F_1^e

tid	Name	Race	Job	SSN^1	INS^1
1	~	~	~	$enc(SSN, K^1_{SSN})$	$enc(150, K^1_{INS})$
2	~	~	~	$enc(SSN, K^2_{SSN})$	$enc(100, K^2_{INS})$
3	~	~	~	$enc(SSN, K^3_{SSN})$	$enc(100, K^3_{INS})$

F_2^e

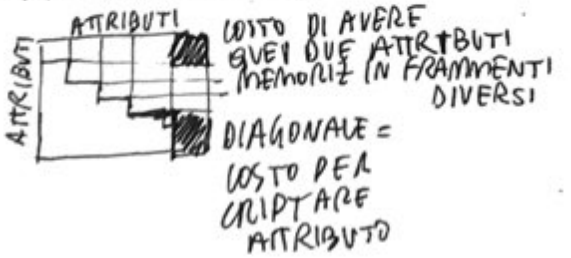
tid	Disease	Treat	SSN^2	INS^2
1	~	~	K^1_{SSN}	K^1_{INS}
2	~	~	K^2_{SSN}	K^2_{INS}
3	~	~	K^3_{SSN}	K^3_{INS}

(b) FRAGMENTATION METRICS

CI POTREBBERO ESSERE PIU' FRAMMENTAZIONI CHE SODDISFANO I CC. SERVE UNA METRICA PER MISURARE LA QUALITA' DI UNA FRAMMENTAZIONE, IN BASE AL CARICO DI QUERY RICHIESTE ALL'UTENTE PER RECUPERARE I DATI DAI FRAMMENTI.

QUERY WORKLOAD DESCRIVE CON QUANTA FREQ GLI ATTRIBUTI APPAIONO INSIEME NELLE QUERY COSI' DA STIMARE IL CARICO COMPUTAZ SE UNA FRAMMENTAZIONE DOVESSE DIVIDERE QUEGLI ATTRIBUTI.

MATRICE DI AFFINITA'



IL COSTO DI UNA FRAMMENTAZIONE E' CALCOLATO SOMMANDO I COSTI DEGLI ATTRIBUTI CRIPTATI E I COSTI DELLE COPPIE DI ATTRIBUTI NON MEMORIZZATI INSIEME.

(c) COMPUTING AN OPTIMAL FRAGMENTATION

IL CALCOLO DI UNA FRAMMENTAZIONE CHE MINIMIZZI I COSTI E' NP-HARD.

COSTRUIAMO UN IPERGRAFO
- VERTICI ATTRIBUTI
- VERTICI E INDICI SONO PESATI IN BASE AL VALORE DELLA CELLA DELLA MATRICE DI AFFINITA'
- GLI IPERARCHI MODELLANO I CC

USIAMO UN APPROCCIO EURISTICO CON DUE TECNICHE DI APPROSSIMAZIONE CHE USATE INSIEME PERMETTONO IL CALCOLO IN TEMPO POLINOMIALE:

- MIN CUT CALCOLO FRAMMENTO MINIMO = CALCOLO TAGLIO MINIMO

  UN TAGLIO MINIMO E' UN PARTIZIONAMENTO DEL GRUPPO DI VERTICI IN DUE SOTTOGRUPPI DI VERTICI (V1 E V2) CHE MINIMIZZANO IL PESO DEGLI ARCHI (CON UN VERTICE IN V1 E L'ALTRO IN V2).

  CI SONO TANTI TAGLI E SCEGLIAMO QUELLO CHE SODDISFA PIU' CC.

- WEIGTHED SET COVER

  QUANDO NON CONSIDERIAMO IL COSTO DI DIVISIONE DEGLI ATTRIBUTI IN FRAMMENTI.

  CALCOLARE IL FRAMMENTO MINIMO = MINIMUM SET COVER PROBLEM

  IL MINIMUM SET COVER E' IL SET DI ATTRIBUTI CON IL PESO MINIMO CHE INCLUDE ALMENO UN ATTRIBUTO PER OGNI CONSTRAINT.

(d) QUERY EVALUATION

LA QUERY ORIGINALE VIENE TRADOTTA IN UNA SERIE DI QUERY CHE OPERANO SUI DUE FRAMMENTI.

LA RISOLUZIONE DELLE QUERY PUO' AVVENIRE

- IN PARALLELO
  I DUE PROVIDERS RISOLVONO q_1 E q_2 IN PARALLELO. USER FA IL JOIN E POI DECRIPTA.
- IN SERIE
  UNO DEI DUE PROVIDER FA LA QUERY E USER INVIA ALL'ALTRO PROVIDER SOLO tid RISULTANTI DA q_1. USER FA IL JOIN.

ESEMPIO IN PARALLELO

q: SELECT Att
FROM R
WHERE Cond

q_1: SELECT tid, Att
FROM F_1^e
WHERE Cond1

q_2: SELECT tid, Att
FROM F_2^e
WHERE Cond2

q_u: SELECT Att
FROM R1 JOIN R2
ON R1.tid = R2.tid
WHERE Cond u

q: SELECT Name
FROM Patients
WHERE Job = Lawyer
Disease = flu
Ins = 100

q_1: SELECT tid, Name, Ins^1
FROM F_1^e
WHERE Job = Lawyer

q_2: SELECT tid, Ins^2
FROM F_2^e
WHERE Disease = flu

q_u: SELECT Name
FROM R1 JOIN R2
ON R1.tid = R2.tid
WHERE Decrypt (INS1, INS2') = 100

ESEMPIO IN SERIE

q: SELECT Att
FROM R
WHERE Cond

q_1: SELECT tid, Att
FROM F_1^e
WHERE Cond1

q_2: SELECT tid, Att
FROM F_2^e
WHERE (tid IN R1.tid)
AND Cond2

q_u: SELECT Att
FROM R1 JOIN R2
ON R1.tid = R2.tid
WHERE Condu

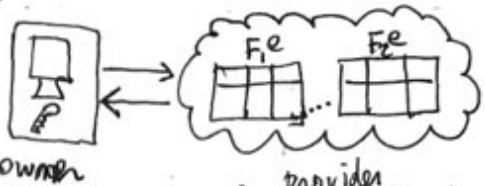
q: SELECT Name
FROM Patients
WHERE Job = Lawyer
Disease = flu
Ins = 100

q_1: SELECT tid, Name, Ins^1
FROM F_1^e
WHERE Job = Lawyer

q_2: SELECT tid, Ins^2
FROM F_2^e tid q1
WHERE (tid IN {4,6})
AND Disease = flu

q_u: SELECT Name
FROM R1 JOIN R2
ON R1.tid = R2.tid
WHERE Decrypt (Ins^1, Ins^2) = 100

③ MULTIPLE FRAGMENTS



owner provider

MEMORIZZO SU PIU' FRAMMENTI. SENSIBILI
VENGONO CRIPTATI GLI ATTRIBUTI
E LE ASSOCIAZIONI SENSIBILI.
I FRAMMENTI NON SONO LINKABILI (PERCHÉ NON HANNO ATTRIBUTI IN COMUNE) E SONO COMPLETI (TUTTI GLI ATTRIBUTI SONO MEMORIZ IN OGNI FRAMMENTO).

④ FRAGMENTATION MODEL

I DATI SONO MEMORIZZATI IN UN NUMERO VARIABILE DI FRAMMENTI (ANCHE DELLO STESSO PROVIDER). I FRAMMENTI SONO LINKABILI SOLO DALL'OWNER.
LA FRAMMENTAZIONE VERTICALE GARANTISCE RISPETTO DEI CC.

CONDIZIONI:
- UN FRAMMENTO NON PUO' CONTENERE TUTTI GLI ATTRIBUTI CHE COMPONGONO UN CC.
- I FRAMMENTI NON POSSONO ESSERE LINKABILI

UNA FRAMMENTAZIONE MASSIMIZZA LA VISIBILITA' SE OGNI ATTRIBUTO NELLA TAB (CHE NON E' IN UN SINGLETON CONSTRAINT) E' INCHIARO IN ALMENO UN FRAMMENTO.

SI NOTI CHE PER SODDISFARE LA CONDIZIONE DI NON LINKABILITY, OGNI ATTRIBUTO CHE NON APPARE IN UN SINGLETON CONSTRAINT PUO' APPARTENERE AL MASSIMO A UN FRAMMENTO.

ESEMPIO
F_1 = {Name, Job}
F_2 = {Disease, Treatment}
F_3 = {Race, Ins}

MASSIMIZZO LA VISIBILITA' PERCHÉ TUTTI GLI ATTRIBUTI NON CONSTRAINT SONO IN CHIARO IN UN SOLO FRAMMENTO

FRAMMENTO F_i^e E' COMPOSTO DA
- salt VALORE RANDOM CHE FA DA PK
- enc ATTRIBUTO CRIPTATO CON IL SALE
- Attributi

salt	enc	Name	Job
S_1^1	~	Alice	teacher
S_2^1	~	Bob	farmer
S_3^1	~	Carol	nurse

F_1^e

salt	enc	disease	treatment
S_1^2	~	flu	panac
S_2^2	~	asthma	bronco
S_3^2	~	gastritis	antiac

F_2^e

salt	enc	Race	Ins
S_1^3	~	white	160
S_2^3	~	white	100
S_3^3	~	asian	100

F_3^e

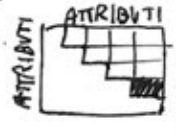
(b) FRAGMENTATION METRICS

CI POTREBBERO ESSERE PIU' FRAMMENTI CHE SODDISFANO MASSIMA VISIBILITA'.

METRICHE BASATE SUL CARICO COMPUTAZIONALE DELLE QUERY:

- MINIMAL FRAGMENTATION
  OBIETTIVO: MINIMIZZARE NUMERO DI FRAMMENTI.
  COSI' DA AUMENTARE IL NUMERO DI ATTRIBUTI PER FRAMMENTO E DIMINUIRE IL CARICO COMPUTAZIONALE DELLE QUERY.
- MAXIMUM AFFINITY
  L'AFFINITA' TRA DUE ATTRIBUTI QUANTIFICA I VANTAGGI DI PERFORMANCE DELLA QUERY SE QUESTI VENISSERO MEMORIZ NELLO STESSO FRAMMENTO.
  PERCHE' DUE ATTRIBUTI CON ALTA AFFINITA' SONO PIU' FREQ NELLA STESSA QUERY.

MATRICE AFFINITA'



BENEFICIO DI MEMORIZ INSIEME I 2 ATTRIBUTI

! NO GLI ATTRIBUTI CHE SONO SINGLETON CONSTRAINT (COME SSN)

LA QUALITA' DI UNA FRAMMENTAZIONE E' CALCOLATA COME LA SOMMA DELLE AFFINITA' TRA I FRAMMENTI CHE LA COMPONGONO.

- MINIMUM QUERY EVALUATION COST

PRENDE IN CONSIDERAZIONE IL COSTO DI UN GRUPPO DI QUERY RAPPRESENTATIVE.
RISPETTO ALLA MATRICE DI AFFINITA' PRENDE IN CONSIDERAZIONE ANCHE I BENEFICI DI MEMORIZZAZIONE NELLO STESSO FRAMMENTO DI UN GRUPPO A CASO DI ATTRIBUTI.

IL COSTO DELLA QUERY E' CALCOLATO SULLA BASE DEL PESO DEL RISULTATO CHE RICEVE L'OWNER POICHE' IL CALCOLO LATO CLIENT E' IL PIU' ONEROSO E VOGLIAMO RIDURLO AL MINIMO.

(c) COMPUTING AN OPTIMAL FRAGMENTATION

- NP-HARD A PRESCINDERE DALLA METRICA
- POLINOMIALE TRAMITE COLORAZIONE IPERGRAFO

(d) QUERY EVALUATION

OVVIAMENTE L'IDEALE E' FARE QUERY SU FRAMMENTI CON ATTRIBUTI IN CHIARO.

COSA SUCCEDE CON ATTRIBUTI CRIPTATI:
- SCARICO FRAMMENTO DAL CLOUD
- DECRIPTO
- CALCOLO q IN LOCALE

ESEMPIO

q: SELECT Att FROM R WHERE Cond

qp: SELECT salt, enc, Att FROM Fe WHERE Condp

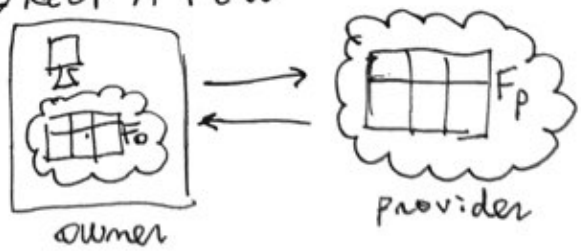
qv: SELECT Att FROM Decrypt(tabp.enc ⊕ salt, k) WHERE Condv

q: SELECT Name FROM Patient WHERE Disease=flu Job=teacher

qp: SELECT salt, enc, Name FROM Fe WHERE Job=teacher

qv: SELECT Name FROM Decrypt(tabp.enc ⊕ salt, k) WHERE Disease=flu

(a) KEEP A FEW



OWNER (trusted) MEMORIZZA UNA PICCOLA QUANTITÀ DI DATI SENZA CRIPTARLI. GLI ATTRIBUTI SENSIBILI SONO SALVATI NELL'OWNER. LE ASSOCIAZIONI SENSIBILI SONO PROTETTE SALVANDO ALMENO UN ATTRIBUTO NELL'OWNER.
OBIETTIVO: MINIMIZZARE CARICO OWNER.

(a) FRAGMENTATION MODEL

[vedi sopra]
FRAMMENTAZIONE CORRETTA QUANDO:
- FRAMMENTO NEL PROVIDER NON PUO' CONTENERE TUTTI GLI ATTRIBUTI CHE COMPONGONO UN CC
- TUTTI GLI ATTRIBUTI SONO MEMORIZZATI LATO OWNER E PROVIDER MA RIDONDANZA LATO OWNER POTREBBE ESSERE NON NECESSARIA E TROPPO COSTOSA
- I FRAMMENTI NON SONO LINKABILI

ESEMPIO $F_o = \{SSN, Name, Ins\}$
$F_p = \{Race, Job, Disease, treatment\}$

F_o E F_p HANNO UN ATTRIBUTO CHIAVE PER POTER RICOSTRUIRE TAB ORIGINALE. QUESTO ATTRIBUTO PUO' ESSERE LA PK DELLA TAB ORIGINALE (SE NON SENSIBILE) O UN tid.

Tid	SSN	Name	Ins
1	123..	Alice	160
2	123...	Bob	100
3	123...	Carol	100

F_o^e

Tid	Race	Job	Disease	treatment
1	white	teacher	flu	panac
2	white	farmer	asthma	bronco
3	asian	nurse	gastrite	antiac

F_p^e

(b) FRAGMENTATION METRICS

POTREBBERO ESISTERE DIFFERENTI FRAMM CORRETTE. OWNER VUOLE MEMORIZZARE MENO POSSIBILE, PER FARE QUESTO SERVE CALCOLARE CARICHI CAUSATI DA F_o.

METRICHE:
- MINIMAL FRAGMENTATION (CARDINALITY)
SI CONTANO GLI ATTRIBUTI IN F_o PERCHÉ UN FRAMMENTO CON POCHI ATTRIBUTI È PIU' PROB CHE SIA PICCOLO E CHE SIA COINVOLTO IN POCHE QUERY.
- MINIMAL SIZE OF ATTRIBUTES (STORAGE)
OBIETTIVO È 'MINIMIZZARE LO STORAGE IN F_o CALCOLATO COME LA SOMMA DELLA DIMENSIONE DEGLI ATTRIBUTI.
- MINIMAL NUMBER OF QUERY (COMPUTATION)
IL CARICO COMPUTAZIONALE È CALCOLATO IN BASE AL NUMERO DI QUERY CHE INVOLVONO OWNER PER ESSERE VALUTATE. (CIOE' TUTTE QUELLE QUERY CHE INVOLVONO ALMENO UN ATTRIBUTO IN F_o).
SERVE CONOSCERE IL QUERY WORKLOAD:
IL COSTO DELLA FRAMMENTAZIONE È CALCOLATO COME LA SOMMA DELLE FREQUENZE DI QUELLE QUERY CHE RICHIEDONO ALMENO UN ATTRIBUTO DI F_o.
- MINIMAL NUMBER OF CONDITIONS (FREQUENCY)
IL CARICO COMPUTAZIONALE DELL'OWNER È DATO DAL NUMERO DI CONDIZIONI CHE L'OWNER DOVREBBE VALUTARE, POICHÉ MOLTE CONDIZIONI NELLA STESSA QUERY CAUSANO UN CARICO COMPUTAZ MAGGIORE.

COSTO FRAMMENTAZIONE È LA SOMMA DELLE FREQUENZE DELLE CONDIZIONI IN q CHE INVOLVONO ATTRIBUTI IN F_o.

(c) COMPUTING AN OPTIMAL FRAGMENTATION
- NP-HARD
- POLINOMIALE CON MINIMUM HITING SET
- APPROCCIO EURISTICO:
ALGORITMO CALCOLA LA FRAMMENTAZIONE MINIMA LOCALE, CHE È UNA FRAM DOVE NESSUN ATTRIBUTO PUO' ESSERE SPOSTATO DA F_o A F_p SENZA VIOLARE UN CC.

(d) QUERY EVALUATION

LA QUERY ORIGINALE VA TRADOTTA IN q_o E q_p. LA VALUTAZIONE DELLA QUERY q PUO' ESSERE FATTA IN DUE MODI:

- PROVIDER → OWNER
$Cond_p \rightarrow Cond_o$ AND $Cond_{po}$

q: SELECT Att FROM R WHERE Cond

q_p: SELECT tid, Att FROM F_p^e WHERE $Cond_p$

q_o: SELECT Att FROM F_o^e JOIN R_p ON F_o^e.tid = R_p.tid WHERE $Cond_o$ AND $Cond_{po}$

- OWNER → PROVIDER
$Cond_o \rightarrow Cond_p \rightarrow Cond_{po}$

q: SELECT Att FROM R WHERE Cond

q_o: SELECT tid FROM F_o^e WHERE $Cond_o$

q_{po}: SELECT Att FROM F_o^e JOIN R_p ON F_o^e.tid = R_p.tid WHERE $Cond_{po}$

q_p: SELECT tid, Att FROM F_p^e WHERE (tid IN R_o) AND $Cond_p$

SELECTIVE ACCESS

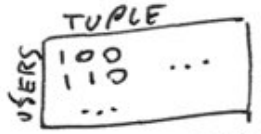
L'ENCRYPTION È UNA TECNICA PER PROTEGGERE I DATI MA COMPLICA QUERY E ACCESSO AI DATI.

I PROBLEMI DI CRIPTARE I DATI FUORISONO:
- LA CONSEGUENZA DI CRIPTARE I DATI CON UNA SOLA CHIAVE È CHE TUTTI GLI UTENTI POSSONO VEDERE TUTTO
- QUERY SONO PIU' COMPLICATE PERCHÉ PROVIDER NON PUO' SVOLGERLE DIRETTAM SUL CRIPTATO

SELECTIVE ENCRYPTION

DIFFERENTI DATI SONO CRIPTATI CON DIFFERENTI CHIAVI IN BASE A CHI PUO' AVERE ACCESSO. OGNI UTENTE PUO' DECRIPTARE E ACCEDERE A UN SUBSET DI TUPLE IN BASE ALLE CHIAVI CHE CONOSCE.

OWNER DECIDE LA POLITICA DI AUTORIZZAZIONE



MATRICE DI ACCESSO

LA POLITICA DI ENCRYPTION DEFINISCE E REGOLA IL MAZZO DI CHIAVI USATO PER CRIPTARE LE TUPLE E REGOLA LA DISTRIBUZ DELLE CHIAVI AGLI UTENTI.

POLITICA ENCRYPTION = POLITICA AUTORIZZAZIONE

LA TRADUZIONE DELLA POLITICA AUTORIZZAZIONE IN UNA POLITICA DI ENCRYPTION DEVE GARANTIRE CHE
- OGNI UTENTE DEVE GESTIRE UNA SOLA CHIAVE
- OGNI TUPLA È CRIPTATA CON UNA SOLA CHIAVE

USO UNA TECNICA DI DERIVAZIONE DELLE CHIAVI CHE PERMETTE DI COMPUTARE UNA CHIAVE DI ENCRYPTION K_j PARTENDO DA UNA GIA' CONOSCIUTA CHIAVE K_i PIU' UNA INFORMAZIONE PUBBLICA.

PER DETERMINARE QUALE CHIAVE PUO' ESSERE DERIVATA DA QUALE CHIAVE SERVE UNA GERARCHIA DI DERIVAZIONE DELLE CHIAVI (da qui il termine CRITTOGRAFIA GERARCHICA) CHE PUO' ESSERE RAPPRESENTATA GRAFICAM CON UN GRAFO DIRETTO DOVE
- VERTICE V_i INDICA LA CHIAVE K_i
- ARCO (V_i, V_j) INDICA UN COLLEGAMENTO TRA LA CHIAVE K_i E LA CHIAVE K_j E SIGNIFICA CHE K_j PUO' ESSERE DERIVATA DA K_i.

LA DERIVAZIONE DELLE CHIAVI E' UNA CATENA: K_j PUO' ESSERE CALCOLATA A PARTIRE DA K_i SE C'E' UN PATH DA V_i A V_j NELLA GERARCHIA.

LA DERIVAZIONE DI CHIAVI BASATA SU TOKEN (TOKEN-BASED KEY DERIVATION) È UNA FORMA DI GERARCHIA CHE MINIMIZZA IL BISOGNO DI CRIPTARE E/O REDISTRIBUIRE LE CHIAVI NEL CASO DI AGGIORNAMENTO DELLA POLITICA DI AUTORIZZAZIONE.

l: etichetta
K: chiave

GRAZIE AL TOKEN i PUO' DERIVARE j.
I TOKEN SONO PUBBLICI.

$K_1 l_1$ — $K_8 l_8$ — $K_{10} l_{10}$

SE CONOSCO K_1 POSSO DERIVARE K_8 K_{10}

FUNZIONE $\phi: U \cup R \to L$

FUNZIONE ϕ MAPPA GLI UTENTI U E LE RISORSE R NELLE ETICHETTE L DELLE CHIAVI.

POLITICA DI CRIPTAZIONE

U UTENTI	R RISORSE
K CHIAVI	L ETICHETTE
ϕ POLITICA DI ASSEGNAMENTO DELLE CHIAVI AGLI UTENTI E ALLE RISORSE	T TOKEN CHE PERMETTONO LA DERIVAZIONE

$A \xrightarrow{\phi} K_1 l_1 \xrightarrow{\text{token}} K_8 l_8 \xrightarrow{\phi} r_1$
$K_1 l_1 \xrightarrow{\text{token}} K_{10} l_{10} \dashrightarrow r_2$

user A con access $\{r_1, r_2\}$
r_1 CRIPTATA CON K_8, r_2 con K_{10}

DEFINIAMO UNA GERARCHIA DI DERIVAZIONE DELLE CHIAVI ADATTA AL CONTROLLO DELL ACCESSO E ADATTA ALLA GESTIONE DELLE CHIAVI. ABBIAMO UN DAG
- VERTICE: OGNI ELEMENTO DELL'INSIEME DEGLI UTENTI U

· CAMMINO DA v_i A v_j SE IL SET DI UTENTI RAPPRESENTATO DA v_i E' UN SUBSET DI QUELLO RAPPRESENTATO DA v_j.

POLITICA DI AUTORIZZAZIONE E' GARANTITA SE
- AD OGNI UTENTE U_i VIENE COMUNICATA LA CHIAVE ASSOCIATA AL SUO VERTICE
- OGNI TUPLA t_j E' CRIPTATA CON LA CHIAVE DEL VERTICE CHE RAPPRESENTA $ACL(t_j)$ SPIEGARE MEGLIO

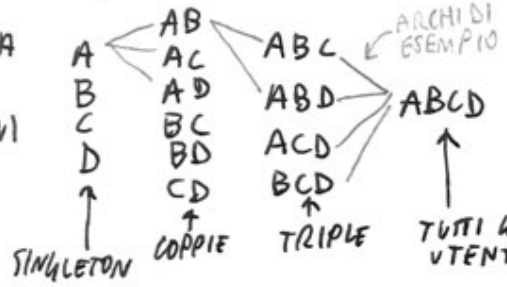
IN QUESTO MODO OGNI TUPLA PUO' ESSERE DECRIPTATA E ACCEDUTA SOLO DAGLI UTENTI NELLA SUA ACL.

ESEMPIO

MATRICE ACCESSO

	t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8
A	1	1	0	1	1	1	1	0
B	1	1	1	1	1	0	0	0
C	1	1	1	0	1	1	0	0
D	0	0	0	1	1	1	0	1

GERARCHIA DI DERIV DELLE CHIAVI



CHIAVI ASSEGNATE ALL'UTENTE

A	K_A
B	K_B
C	K_C
D	K_D

CHIAVI USATE PER CRIPTARE

t_1	K_{ABC}
t_2	K_{ABC}
t_3	K_{BC}
t_4	K_{ABD}
t_5	K_{ABCD}
t_6	K_{ACD}
t_7	K_A
t_8	K_D

PERO' QUESTO APPROCCIO CREA PIU' CHIAVI E PIU' TOKEN DEL NECESSARIO.

MINIMIZZARE I TOKEN E' NP-HARD.

SOLUZIONE EURISTICA. PREMESSE:
- I VERTICI CHE SERVONO PER APPLICARE CORRETTAMENTE UNA POLITICA DI AUTORIZ SONO SOLO QUELLI CHE RAPPRESENTANO UN SINGLETON GRUPPO DI UTENTI (= CHIAVI DEGLI UTENTI) E L'ACL DELLE TUPLE (= CHIAVI PER CRIPTARE I FILE)
- QUANDO DUE O PIU' VERTICI HANNO PIU' DI DUE ANTENATI COMUNI, INSERIMENTO DI UN VERTICE CHE RAPPRESENTA QUESTO GRUPPO DI UTENTI RIDUCE IL NUMERO TOTALE DI TOKENS

APPROCCIO EURISTICO: (PER TROVARE LA POLITICA DI ENCRYPTION MINIMA)

1. INITIALIZATION

ALGORITMO IDENTIFICA I VERTICI NECESSARI PER IMPLEMENTARE LA POLITICA DI AUTORIZZAZIONE CIOE' QUEI VERTICI CHE RAPPRESENTANO
- SINGLETON
  GRUPPI DI UTENTI SINGLETON LE CUI CHIAVI SONO COMUNICATE AGLI UTENTI (E QUESTO PERMETTE LORO DI DERIVARE LE CHIAVI DI QUELLE TUPLE A CUI POSSONO ACCEDERE)
- NON SINGLETON IN BASE ALL'ACL
  L'ACL DELLE TUPLE LE CUI CHIAVI SONO USATE PER CRIPTARE

	R_1	R_2	R_3	R_4	R_5
A	0	1	0	1	1
B	1	1	1	1	1
C	0	1	1	1	1
D	0	0	1	1	1
	α	B	γ	$\in$	$\in$

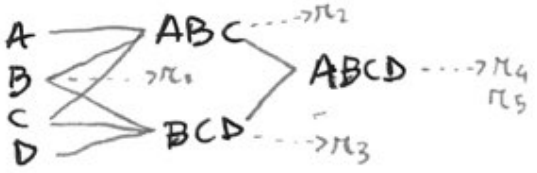
A ABC 3
B
C
D BCD γ

ABCD $\in$

2. COVERING

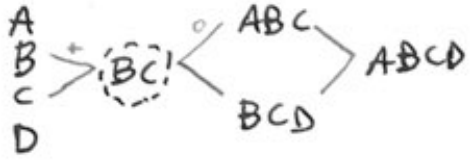
PER OGNI VERTICE V (CHE CORRISPONDE A UN INSIEME DI UTENTI NON SINGLETON) TROVO UN COVER CHE COPRA L'INSIEME.
ALGO TROVA UN GRUPPO DI VERTICI CHE FORMA UN NON REDUNDANT SET COVERING PER V.
SIGNIFICA ANDARE A VEDERE IL GRUPPO DI SOTTOINSIEMI LA CUI UNIONE MI DA' L'INSIEME.

ESEMPIO: SE PRENDO ABCD DEVO TROVARE UN GRUPPO DI SOTTOINSIEMI LA CUI UNIONE MI DA' ABCD.
FACCIO QUESTO RAGIONAMENTO AD OGNI LIVELLO (DAL PIU' GRANDE AI SINGLETON).
NELLA PRATICA SIGNIFICA ASSICURARSI CHE A, B, C E D RIESCANO AD ARRIVARE A ABCD.

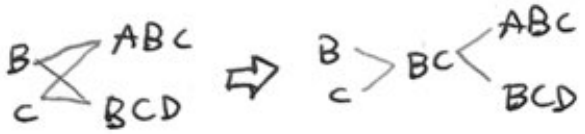


3. FACTORIZATION

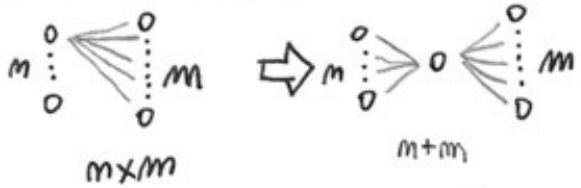
PER OGNI GRUPPO DI VERTICI $V_1 \dots V_m$ CHE HANNO PIU' DI $m > 2$ ANTENATI COMUNI $V_1 \dots V_m$, ALGORITMO INSERISCE UN VERTICE INTERMEDIO V CHE RAPPRESENTA TUTTI GLI UTENTI IN $V_1 \dots V_m$ E CONNETTE OGNI V_i CON V_* E V CON OGNI V_j.



SOSTANZIALMENTE SI CREA UN HUB:



COSI' DA PASSARE DA $m \cdot m$ A $m+m$:



AGGIORNARE ACL (SEL/BEL)

ESISTONO DUE STRATI DI ENCRYPTION OGNUNO CON LA PROPRIA POLICY COSI' DA DELEGARE PARZIALMENTE AL SERVER LA GESTIONE DEI GRANT/REVOKE.

- BASE ENCRYPTION LAYER (BEL)

E' APPLICATO DALL'OWNER PRIMA DI METTERE FUORI I DATI. LA GERARCHIA BEL DI DERIVAZIONE DELLE CHIAVI E' COSTRUITA DALLA POLITICA DI AUTORIZZAZIONE.

IN CASO DI AGGIORNAMENTO DELLA POLITICA, BEL È AGGIORNATO INSERENDO IL TOKEN NEL CATALOGO PUBBLICO.

OGNI VERTICE NELLA GERARCHIA BEL HA 2 CHIAVI
- CHIAVE DI DERIVAZIONE K
- CHIAVE DI ACCESSO K_A (PER DECRIPTARE TUPLE)

K_A E' UN HASH DI K

- SURFACE ENCRYPTION LAYER (SEL)

E' APPLICATO DAL SERVER SULLE TUPLE GIA' CRIPTATE DAL BEL. APPLICA GLI AGGIORNAM DELLA POLITICA DI AUTORIZZAZIONE RECRIPTANDO LE TUPLE E CAMBIANDO LA GERARCHIA BEL. I VERTICI DELLA GERARCHIA SEL SONO ASSOCIATI CON UNA SOLA CHIAVE K^S.

- OPERATIONS

UN UTENTE PUO' ACCEDERE A UNA TUPLA SOLO SE CONOSCE LE CHIAVI USATE PER CRIPTARE LA TUPLA A LIVELLO BEL E SEL.

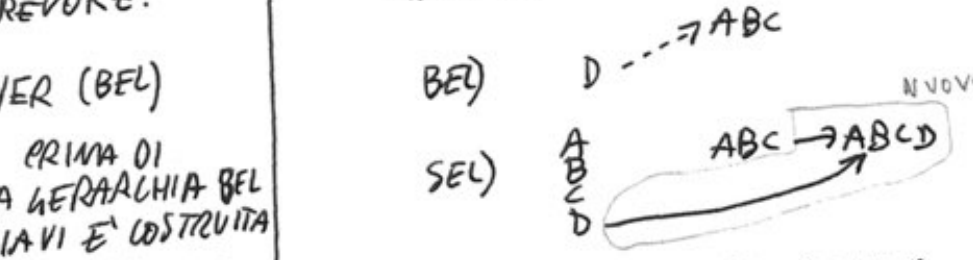
- GRANT QUANDO A UN UTENTE VIENE DATA L'AUTORIZZAZIONE PER ACCEDERE A UNA TUPLA, DEVE CONOSCERE LE CHIAVI USATE PER CRIPTARE LA TUPLA AI LIVELLI SEL E BEL.

  OWNER AGGIUNGE UN TOKEN NELLA GERARCHIA BEL DAL VERTICE CHE RAPPRESENTA L'UTENTE AL VERTICE LE CUI CHIAVI SONO STATE USATE PER CRIPTARE LA TUPLA.

  OWNER CHIEDE AL SERVER DI AGGIORNARE LA SUA GERARCHIA E RECRIPTARE LA TUPLA. (ANCHE ALTRE TUPLE POTREBBERO AVER BISOGNO DI ESSERE RECRIPTATE)

ESEMPIO:

GRANT D PUO' ACCEDERE A t_1:ABC



!SEL DOVRA' FARE UNA OVER ENCRYPTION SE ABCD SI COLLEGA AD ALTRE TUPLE A CUI D NON PUO' ACCEDERE

- REVOKE QUANDO UN UTENTE PERDE I PRIVILEGI DI ACCESSO A UNA TUPLA, OWNER CHIEDE AL SERVER DI RECRIPTARE LA TUPLA A LIVELLO SEL.

  SE IL VERTICE DEGLI UTENTI CHE RICEVONO LA REVOKE NON E' RAPPRESENTATO NELLA GERARCHIA, ALLORA SERVER
  - INSERISCE NUOVO VERTICE
  - RECRIPTA LE TUPLE

ESEMPIO:

REVOKE B NON PUO' PIU' ACCEDERE A t_4:ABDE

BEL) non succede nulla

SEL)

t_4: K^SABDE $\xrightarrow[\text{diventa}]{}$ t_4 : K^SADE

COLLUSIONI

- IL SISTEMA BEL+SEL E' VULNERABILE A COLLUSIONI CIOE' QUANDO PIU' UTENTI SI METTONO D'ACCORDO PER AVERE PIU' INFO. CASISTICA: PIU' UTENTI SI SCAMBIANO LE CHIAVI CON IL SERVER E RIESCONO AD ACCEDERE A QUALCOSA CHE DA SOLI NON AVREBBERO POTUTO.

4 CASI

PRIVACY QUERY

LE QUERY FATTE A UN SERVER POSSONO ESSERE SCOPERTE DAL SERVER STESSO METTENDO A RISCHIO LA PRIVACY DEGLI UTENTI E LA PRIVACY DEI DATI.
MONITORANDO I PATTERNS DI ACCESSO ALLE TUPLE UN ATTACCANTE PUO' INFERIRE QUALCOSA SUL VALORE DELLE TUPLE.

E' NECESSARIO PROTEGGERE SIA LA CONFIDENZIALITA' DELL'ACCESSO (LA SINGOLA QUERY) SIA LA CONFIDENZIALITA' DEL PATTERN (DUE QUERY PUNTANO ALLO STESSO TARGET).

~~ORAM~~

~~IL DATABASE E' ORGANIZZATO IN UN GRUPPO DI M BLOCCHI CRIPTATI MEMORIZZATI A PIRAMIDE. OGNI LIVELLO L MEMORIZZA 4^L BLOCCHI.~~
~~UN BLOOM~~

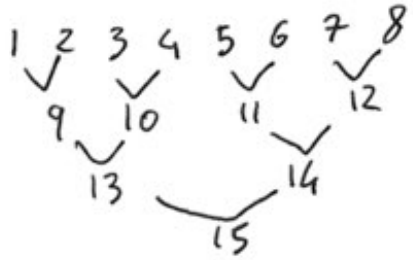
PATH-ORAM

STRUTTURA DATI AD ALBERO I CUI NODI SONO BUCKETS CHE MEMORIZZANO UN NUMERO FISSO DI BLOCCHI (CONTENENTI DATI VERI O FAKE). OGNI BLOCCO E' ASSEGNATO A UNA FOGLIA RANDOM E VIENE MEMORIZZATO LATO CLIENT IN UNA CACHE LOCALE CHIAMATA STASH O IN UNO DEI BUCKET LUNGO IL CAMMINO VERSO LA FOGLIA A CUI E' ASSOCIATO.

LE OPERAZIONI DI LETTURA SCARICANO DAL SERVER E MEMORIZZANO NELLO STASH TUTTI I BUCKETS CHE SI TROVANO NEL PERCORSO DALLA ROOT ALLA FOGLIA. IL TARGET_BLOCK VIENE POI MAPPATO A UNA NUOVA FOGLIA RANDOM.

IL PATH LETTO DAL SERVER VIENE RISCRITTO INSERENDO NEI BUCKETS I BLOCCHI PRESENTI NELLO STASH PERCHE' LO STASH VA SVUOTATO OGNI VOLTA CHE SI SEGUE UN PERCORSO.

ESEMPIO:



STASH |a|b| | | |

DIZIONARIO: position[a]=4 (a SI TROVA LUNGO IL PATH DA 4 A 15.)
per tutte le chiavi

~~LETTURA: OGNI VOLTA CHE LEGGO DEVO RIMAPPARE~~ 12

OGNI VOLTA CHE FACCIO UNA LETTURA DEVO SCARICARE LO STASH.
LEGGO position[c]=7
QUINDI VADO DA ROOT(15) A 7.
SCARICO LO STASH:
- a=4 QUINDI LASCIO a IN 15 PERCHE' E' IL NODO PIU' BASSO IN COMUNE
- b=5 LO POSSO METTERE IN 15 O 14. SCELGO 14 PERCHE' PIU' BASSO

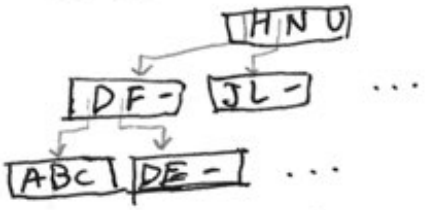
LOSE NEGATIVE DI PATH-ORAM:
- SI RIEMPIONO I NODI VICINO ALLA ROOT
- OGNI LETTURA DIVENTA ANCHE SCRITTURA
- LE LETTURE SONO RITARDATE PERCHE' NON SCRIVO SUBITO

SHUFFLE INDEX

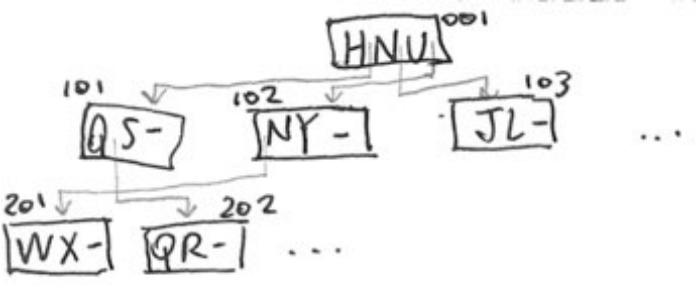
A. DATA STRUCTURE

TECNICA DI INDEXING PER ORGANIZZARE I DATI IN STORAGE E PER ESEGUIRE QUERY IN MODO EFFICIENTE.
HO 3 LIVELLI DI ASTRAZIONE

- LIVELLO ASTRATTO
  ALBERO B+ NON CONCATENATO CON FAN-OUT F (NUMERO DI PUNTATORI AI NODI FIGLI). OGNI NODO RAPPRESENTA LA ROOT DI UN SOTTOALBERO CON ALMENO F/2 FIGLI.
  LE FOGLIE MEMORIZZANO LE TUPLE CON LE LORO KEY VALUE.
  ESSENDO NON CONCATENATE IL SERVER NON RIESCE A SCOPRIRE L'ORDINE DEI VALORI NELLE FOGLIE.



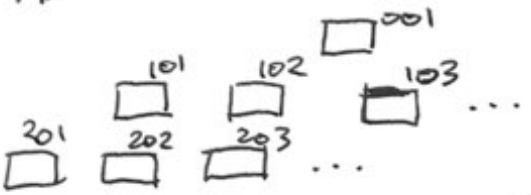
- LIVELLO LOGICO
  OGNI NODO E' RAPPRESENTATO DA UNA COPPIA (id, m) DOVE id E' L'id LOGICO DEL NODO E m E' IL CONTENUTO.
  IL PUNTATORE AL FIGLIO E' RAPPRESENTATO TRAMITE L'id DEL FIGLIO STESSO.



ORDINE NON RIFLETTE LA RELAZIONE VALORE-ORDINE TRA I CONTENUTI DEL NODO.

- LIVELLO FISICO

OGNI NODO (id, m) È CONCATENATO CON UN SALE RANDOM E QUINDI CRIPTATO.
id LOGICO È L'INDIRIZZO FISICO IN CUI È MEMORIZZATO IL BLOCCO.



LA RAPPRESENTAZIONE FISICA COINCIDE CON LA VISIONE CHE IL SERVER HA DEI DATI. INFATTI IL SERVER, OSSERVANDO PER LUNGO TEMPO GLI ACCESSI, PUÒ STABILIRE IN QUALE LIVELLO SI TROVA OGNI BLOCCO.

B. PROTECTION TECHNIQUES

PER PROTEGGERE IL CONTENUTO, L'ACCESSO E I PATTERN CONFIDENTIALITY USIAMO 3 TECNICHE:

- COVER SEARCHES

OBIETTIVO È NASCONDERE IL TARGET ALL'INTERNO DI UN GRUPPO DI TARGET FAKE. COVER SEARCHES SONO RICERCHE FAKE.

PER OGNI LIVELLO DELLO SHUFFLE INDEX IL CLIENT SCARICA NUM-COVER + 1 BLOCCHI (NUM-COVER PER I FAKE; 1 PER QUERY VERA).
PER IL SERVER TUTTI I NUM_COVER+1 BLOCCHI ACCEDUTI HANNO LA STESSA PROB DI AVERE IL TARGET.

COVER SEARCHES DEVONO GARANTIRE:
- SERVER NON DEVE CAPIRE SE UN ACCESSO È VERO O È FAKE
- CAMMINI COVER E VERI DEVONO ESSERE DIVERSI.

- CACHED SEARCHES

OBIETTIVO È PROTEGGERE LE QUERY RIPETUTE RENDENDOLE INDISTINGUIBILI DA QUELLE NON RIPETUTE.
ABBIAMO UNA STRUTTURA CACHE PER LIVELLO. CACHE È LATO CLIENT E MEMORIZZA I NODI LUNGO I PATH DEI TARGET.
QUANTI NODI? GRANDEZZA CACHE m. GLI ULTIMI m ACCEDUTI.

QUANDO IL TARGET DI UN ACCESSO È NELLE CACHE, È SOSTITUITO DA UNA COVER COSÌ DA GARANTIRE CHE NUM-COVER+1 BLOCCHI SIANO SCARICATI PER OGNI LIVELLO.

QUESTO FA SÌ CHE OGNI ACCESSO RIPETUTO SEMBRI UN NUOVO ACCESSO.

ATTENZIONE ATTACCHI OLTRE DIM CACHE.

- SHUFFLING

IDEA: ROMPERE RELAZIONE TRA IL CONTENUTO DEL NODO E IL BLOCCO IN CUI È MEMORIZZATO.

SHUFFLING MUOVE IL CONTENUTO DEI NODI ACCEDUTI (SIA TARGET CHE COVER) E DEI NODI NELLE CACHE. E OGNI VOLTA CHE IL CONTENUTO DI UN NODO È SPOSTATO IN UN ALTRO BLOCCO, IL BLOCCO VIENE RECRIPTATO CON UN NUOVO SALE RANDOM.

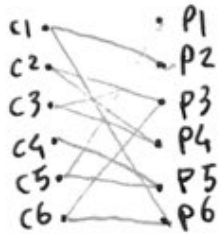
ANONYMIZING BIPARTITE GRAPH

customer	state
c1	NJ
c2	NC
c3	CA
c4	NJ
c5	NC
c6	CA

product	avail
p1	Rx
p2	OTC
p3	OTC
p4	OTC
p5	Rx
p6	OTC

MODELLO LA TABELLA CUSTOMER+PRODUCT COME UN GRAFO BIPARTITO:

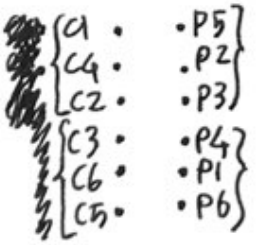
customer	product
c1	p2
c1	p6
c2	p3
c2	p4
c3	p2
c3	p4
c4	p5
c5	p1
c5	p5
c6	p3
c6	p6



NON POSSO PUBBLICARE CUSTOMER+PRODUCT PERCHÉ È SENSIBILE.
PERÒ VOGLIO DARE LA POSSIBILITÀ DI RISPONDERE A CERTE DOMANDE:

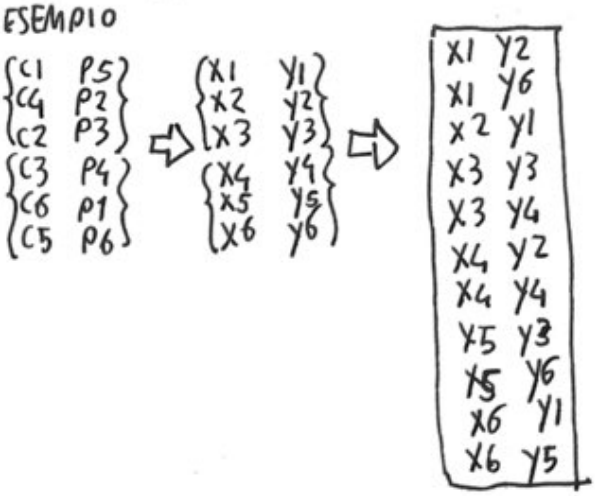
- TIPO 0: STRUTTURA DEL GRAFO
  numero medio di prodotti comprati
- TIPO 1: CONDIZIONE SU UNA PARTE DEL GRAFO
  numero medio di prodotti comprati a Milano
- TIPO 2: CONDIZIONE SU ENTRAMBE LE PARTI DEL GRAFO
  numero medio di prodotti OTC comprati a Milano

→ (k,l)-GROUPING
CREO DEI GRUPPETTI CON K ELEMENTI CHE SI COLLEGANO A L GRUPPI.



COSA PUBBLICO?
GLI ARCHI (CIOÈ LE ASSOCIAZIONI) DI UN GRAFO ISOMORFO RISPETTO ALL'ORIGINALE (CIOÈ UN GRAFO UGUALE MA CON LE ASSOCIAZIONI RINOMINATE)

ESEMPIO



FRAGMENT AND LOOSE ASSOCIATION

UNA FRAMMENTAZIONE È CORRETTA SE
- NON VIOLA I CC
- SODDISFA I VINCOLI DI VISIBILITÀ
- NON MI ESPONE A CORRELAZIONI

SERVE UN TRADEOFF TRA CONFIDENTIALITY CONSTRAINTS E VISIBILITY REQUIREMENTS.
- CONFIDENTIALITY CONSTRAINTS: ALCUNI ATTRIBUTI SONO SENSIBILI NON POSSONO ESSERE RILASCIATI
- VISIBILITY REQUIREMENTS: VISTE SUI DATI

SSN	PATIENT	BIRTH	ZIP	ILLNESS	DOCTOR
~	~	~	~	~	~
~	~	~	~	~	~

C_0: SSN
C_1: PATIENT, ILLNESS
C_2: PATIENT, DOCTOR
C_3: BIRTH, ZIP, ILLNESS
C_4: BIRTH, ZIP, DOCTOR

PUBBLICARE IN MODO LOOSE SIGNIFICA PUBBLICARE IN MODO NON SPECIFICO COSÌ DA GARANTIRE UN CERTO GRADO DI PRIVACY.

LA LOOSE ASSOCIATION NASCONDE LE TUPLE IN GRUPPI E RILASCIA INFORMAZIONI RIGUARDO LE ASSOCIAZIONI SOLO A LIVELLO DI GRUPPO.

COME FACCIAMO A PUBBLICARE LOOSE CON DUE FRAMMENTI Fl E Fr?

• K-GROUPING

UNA FUNZIONE K-GROUPING ASSOCIA OGNI TUPLA IN UN FRAMMENTO A UN GRUPPO IN MODO CHE OGNI GRUPPO ABBIA ALMENO K TUPLE.

K-GROUPING È MINIMALE SE MINIMIZZA LA DIMENSIONE DEL GRUPPO (O SE EQUIVALENTE MASSIMIZZA IL NUMERO DI GRUPPI).

POSSO FARE GROUPING SU DUE FRAMMENTI:
(Kl Kr)-GROUPING SIGNIFICA Kl-GROUPING SU Fl E Kr-GROUPING SU Fr.

(Kl Kr)-GROUPING È MINIMALE SE ENTRAMBI I GROUPING SONO MINIMALI.

ESEMPIO: (2,2)-GROUPING MINIMALE

	BIRTH	ZIP	ILLNESS	DOCTOR
l_1	56/12/9	94142	Hyper	David
l_2	53/3/19	94141	gastrite	Daisy
l_3	58/5/18	94134	Flu	Damian
l_4	53/12/9	94139	asthma	Daniel
l_5	56/12/9	94142	gastritis	Dorothy
l_6	57/6/25	94141	obesity	Drew
l_7	60/7/25	94142	meningitis	Dennis
l_8	53/12/1	94140	hyper	Daisy

OGNI ARCO RAPPRESENTA L'ASSOCIAZIONE TRA UNA TUPLA IN UN GRUPPO IN Fl E UNA TUPLA IN UN GRUPPO IN Fr

LA PROTEZIONE DATA DA K-GROUPING PUÒ ESSERE COMPROMESSA DALLA PRESENZA ALL'INTERNO DI UN GRUPPO DI TUPLE CHE HANNO GLI STESSI VALORI SU ATTRIBUTI LA CUI ASSOCIAZIONE CON ALCUNI ATTRIBUTI NELL'ALTRO FRAMMENTO È SENSIBILE.

ALIKENESS (SOMIGLIANZA)
DUE TUPLE IN Fl (O IN Fr) SONO ALIKE RISPETTO A UN CONSTRAINT c SE QUESTE TUPLE HANNO GLI STESSI VALORI PER GLI ATTRIBUTI DI Fl (O Fr) CHE COMPAIONO IN c.

(l_1 È ALIKE CON l_5 SUL CONSTRAINT C3)

BIRTH
ZIP
ILLNESS

• K-LOOSE ASSOCIATION

UNA ASSOCIAZIONE È K-LOOSE SE PER OGNI GRUPPO g DI NEL FRAMMENTO Fl, L'UNIONE DELLE TUPLE IN TUTTI I GRUPPI CON I QUALI g È ASSOCIATO È UN SET CHE HA CARDINALITÀ ALMENO K E NON CONTIENE DUE TUPLE CHE SONO ALIKE.

= UNA ASSOCIAZIONE È K-LOOSE SE PER OGNI ASSOCIAZIONE DELLA RELAZ ORIGINALE RILASCIA ALMENO K POSSIBILI DISTINTE ASSOCIAZIONI.

ESEMPIO: 4-LOOSE ASSOCIATION

BIRTH	ZIP	G	Gl	Gr	ILLNESS	DOCTOR	G
56/12/9	94142	gz2	gz1	id1	~	~	id1
53/3/19	94141	gz1	gz1	id2	~	~	id1
58/5/18	94139	gz3	gz2	id1	~	~	id2
53/12/9	94139	gz1	gz2	id3	~	~	id2
56/12/9	94142	gz3	gz3	id2	~	~	id4
57/6/25	94141	gz2	gz3	id4	~	~	id3
60/7/25	94142	gz4	gz4	id3	~	~	id3
53/12/1	94140	gz4	gz4	id4	~	~	id4

A

ATTRIBUTO G INDICA A QUALE GRUPPO APPARTIENE OGNI TUPLA.

LA K-LOOSE ASSOCIATION È PUBBLICATA COME UNA RELAZIONE A(Gl, Gr) LE CUI TUPLE SONO COPPIE (gl, gr).

MINIMAL K-LOOSE ASSOCIATION
LA MINIMALITÀ NELL'ASSOCIAZIONE RICHIEDE MINIMALITÀ NEI SINGOLI GRUPPETTI.

DATI
- C CONF CONSTRAINT
- Fl e Fr CON RELATIVE ISTANZE
- PRIVACY DEGREE K

(kl, kr)-GROUPING È MINIMALE SE
- GRUPPO A È K-LOOSE
- NON ESISTE UN GRUPPO K-LOOSE PIÙ PICCOLO

- (K_L, K_R)-GROUPING + K-LOOSE

C'È UNA CORRISPONDENZA TRA IL GRADO DI GROUPING E IL GRADO DI K-LOOSENESS CHE L'ASSOCIAZIONE DI GRUPPO PUO' FORNIRE.

$(K_L\ K_R)$-GROUPING NON PUO' FORNIRE K-LOOSE PER $K > K_L \cdot K_R$.

CI SONO 3 PROPRIETA' DI GROUPING LE CUI SODDISFAZIONI ASSICURANO K-LOOSE CON UN GROUPING MINIMALE:

-ETEROGENEITA' DI GRUPPO

NESSUN GRUPPO PUO' CONTENERE TUPLE CHE SONO ALIKE RISPETTO A C

(2,2)-GROPING NELL'ESEMPIO SODDISFA QUESTA PROPRIETA' POICHÉ TUTTI I GRUPPI IN F_L e F_R HANNO VALORI DIFFERENTI PER GLI ATTRIBUTI C3 E C4.

LA CARDINALITA' DI UN GRUPPO ETEROGENEO FORNISCE UNA MISURA DELLA DIVERSITA' DEL GRUPPO (= NUMERO DI DIFFERENTI VALORI PER GLI ATTRIBUTI NEI CC).

-ETEROGENEITA' DI ASSOCIAZIONE

GARANTISCE CHE L'ASSOCIAZIONE DI GRUPPO NON CONTIENE DUPLICATI. CIOÈ NESSUN GRUPPO PUO' ~~CONTENERE~~ ESSERE ASSOCIATO DUE VOLTE CON LO STESSO GRUPPO.

(2,2)-GROPING NELL'ESEMPIO SODDISFA QUESTA PROPRIETA' POICHÉ ESISTE AL MASSIMO UN ARCO TRA OGNI COPPIA DI GRUPPI.

-ETEROGENEITA' PROFONDA

NESSUN GRUPPO PUO' ESSERE ASSOCIATO CON DUE GRUPPI CHE CONTENGONO TUPLE ALIKE.

(2,2)-GROUPING NELL'ESEMPIO SODDISFA QUESTA PROPRIETA' POICHÉ OGNI GRUPPO IN F_L E' ASSOCIATO CON DUE GRUPPI IN F_R CHE CONTENGONO TUPLE CON DIFFERENTI VALORI Illness/Doctor E Birth/ZIP.

PROPRIETA': QUALUNQUE $(K_L\ K_R)$-GROUP CHE SODDISFA LE 3 PROPRIETA' E CHE HA $K \leq K_L \cdot K_R$ SODDISFA UN DATO GRADO DI K-LOOSE.

ESEMPIO: K-LOOSE DI 12
E' SODDISFATTA CON
(4,3)-GROUPING O
(6,2)-GROUPING O
(12,1)-GROUPING