# Dispensa di Privacy, prof. Pensa

Lorenzo Sciandra

2021-2022

# Leggi sulla privacy

## Conoscenza comune

Intuitivamente possiamo dire che la privacy è legata al concetto di **riservatezza** dell'individuo riguardo ai suoi dati o informazioni personali. Possiamo pensare inerente alla privacy tutto ciò che lede/interferisce nella vita personale. Come conseguenza della privacy possiamo invece pensare al **diritto all'oblio**, l'aver la possibilità di eliminare tutte le informazioni personali passate in modo che non intacchino il presente. Questo diritto è una conseguenza della privacy e in particolare dell'aver il controllo sui propri dati.

## Diverse accezioni di privacy

Nel corso del tempo il termine **privacy** ha assunto diverse accezioni, ognuna delle quali riflette i diversi contesti culturali in cui è stata proposta.

### Privacy = the right to be alone

La privacy venne definita legislativamente nel 1890 da Warren and Brandeis con *"The Right to Privacy"* come il diritto alla riservatezza, all'essere lasciati soli, alla solitudine.

A quel tempo c'era molto potere in mano alle forze dell'ordine che potevano fare in maniera indisturbata spionaggi, percuisizioni e controlli anche senza alcun motivo. La privacy riguardava quindi il diritto di una persona di scegliere l'isolamento dall'attenzione degli altri, il diritto di essere immune da controlli o di essere osservati in contesti privati, come la propria casa. Si parlava dunque di privacy con l'accezione fisica dell'esser indisturbati.

### Privacy = limited access

Possiamo dare alla privacy anche l'accezione di **accesso limitato** alle informazioni personali di un individuo e quindi la possibilità per una persona di partecipare alla società senza che altri individui e organizzazioni raccolgano informazioni su di loro.

Godkin 1880: *"Nulla è meglio tutelare legalmente della propria vita privata, o, in altre parole, il diritto di ogni uomo di tenere per sé i propri affari e di decidere da sé fino a che punto essi debbano essere oggetto di osservazione e discussione pubblica".*

Bok 1989: "*La privacy è la condizione di essere protetti dagli accessi indesiderati da parte di altri, sia fisici, sia riguardo le informazioni personali o l'attenzione*".

### Privacy = control over information

Si va diffondendo nel corso del tempo l'accezione di privacy come **controllo sulle informazioni**.

Fried (1968): "*La privacy non è semplicemente un'assenza di informazioni su di noi nella mente degli altri; piuttosto è il controllo che abbiamo sulle informazioni di noi stessi*".

Westin and Blom-Cooper (1970): "*La privacy è la pretesa di individui, gruppi o istituzioni di determinare autonomamente quando, come e in che misura le informazioni su di loro vengono comunicate ad altri.*"

### Definizione odierna

La privacy entrò poi negli standard internazionali:

- **FIPS PUB 41**: *"Il diritto di un'entità di determinare il grado con cui interagirà con il suo ambiente, compreso il grado in cui l'entità è disposta a condividere le sue informazioni personali con altri"*;
- **ISO**: "*Il diritto delle persone di controllare o influenzare quali informazioni ad esse relative possono essere raccolte e archiviate e da chi e a chi tali informazioni possono essere divulgate.*"

Definizione odierna:

> La privacy è l'abilità di una persona di controllare la disponibilità delle proprie informazioni e la disposizione di se stessa. La possibilità quindi di una persona di poter vivere in maniera **anonima** nella società con eventuali pseudonimi.

## Perchè la privacy è importante?

La privacy è fondamentale per garantire la libertà dell'individuo. Un'assenza di privacy potrebbe portare a discriminazioni nella vita di tutti i giorni per preferenze sessuali, religiose o politiche. E' quindi legata al concetto di **dignità** dell'individuo, non screditare le persone in base ad informazioni personali private presenti e/o passate. La privacy riguarda i sentimenti degli individui che possono infatti essere a *disagio* nel momento in cui delle informazioni che li riguardano vengono diffuse nella comunità o *insicuri* su come le loro informazioni vengano

utilizzate ed in quali ambiti.
La privacy riguarda anche le aziende che possono avere a cuore gli aspetti di privacy dei loro clienti per fidelizzarli al brand e/o per mantenere una buona reputazione. Quindi il compito delle aziende sarà quello di mettere a proprio agio i clienti nel trattamento dei dati personali in modo tale da assumere una posizione di fiducia da parte del cliente. Senza contare il fatto che le aziende devono rispettare delle norme di legge e dovrebbero cercare il più possibile di proteggersi da eventuali dispute legali.

# Tipi di privacy

La privacy può riguardare diversi aspetti:

1. privacy delle idee politiche;
2. privacy del consumatore riguardo alle sue abitudini d'acquisto;
3. privacy medica;
4. privacy degli aspetti biometrici per l'identificazione;
5. proprietà privata;
6. privacy dei record informatici che lasciamo sul web.

Ad oggi, con l'avvento di internet e la digitalizzazione di massa, tutti questi aspetti possono rientrare sotto il cappello di **information/data privacy**.

# Data privacy

Si ha un problema di **data privacy** ogni qual volta ci sono dati, memorizzati in forma digitale o in qualsiasi altra forma, che sono univocamente riconducibili ad una persona o ad un gruppo di persone perfettamente identificabili. Se la persona non è identificabile non abbiamo un problema di privacy, semmai di confidenzialità dei dati.
Il problema di oggi quindi non è **non lasciare dati**, ma evitare che questi siano riconducibili alla persona interessata. Bisogna anche scindere tra dati che possono essere ceduti tranquillamente per ottenere servizi essenziali oggigiorno e dati che vogliamo mantenere privati.
Il diritto alla privacy cambia da paese a paese.

# Leggi e regolamenti sulla privacy

## Stati Uniti

Non esiste un'unica legge che riguarda tutti gli aspetti di privacy. Esistono alcune leggi che regolano alcuni aspetti, ma in maniera superficiale e poco approfondita. Nello specifico non ci sono leggi che riguardano tutta la filiera di raccolta ed uso dei dati. Le aziende, una volta raccolti i dati, possono infatti farne ciò che

desiderano, dato che sono di loro proprietà. Esistono ovviamente alcune eccezioni come ad esempio per i bambini sotto i 13 anni e per dati sanitari.

## Unione Europea

Fortunatamente in unione europea la privacy è molto ben regolamentata:

- L'**articolo 8** della Convenzione Europea dei Diritti dell'Uomo prevede il diritto al rispetto della "*vita privata e familiare, del domicilio e della corrispondenza*".

La Corte Europea dei Diritti dell'Uomo ha dato a questo articolo un'interpretazione molto ampia nella sua giurisprudenza, tra cui: censimenti pubblici, sicurezza nazionale, identificazione personale con impronte digitali e fotografia o raccolta di dati medici o di spesa personale.
La regolamentazione sulla privacy in EU possiamo dividerla in due epoche:

- 1995 - 2018 con la direttiva sulla protezione dei dati;
- 2018 - *oggi* con il GDPR.

Il General Data Protection Regulation si appoggia molto sulla direttiva europea del '95, estendendola.

### Direttiva Europea

In quanto si tratta di una direttiva e non di una legge ogni paese europeo singolarmente poteva implementarla diversamente come meglio preferiva, ovviamente nel rispetto dei seguenti 8 principi di buona pratica. I dati devono essere:

1. Trattati in modo equo e lecito;
2. Processati per scopi limitati;
3. Adeguati, pertinenti e non eccessivi;
4. Accurati;
5. Mantenuti non più del necessario;
6. Trattati nel rispetto dei diritti dell'interessato;
7. Sicuri;
8. Trasferiti solamente a paesi che garantiscano un'opportuna protezione.

Nasce con la direttiva la definizione fondamentale di **dato personale** che è sottostante al concetto di privacy. La sua modifica parziale nel GDPR rispecchia l'evoluzione che ha avuto la società negli anni. Uno degli scopi del GDPR è stato quello di semplificare per il cittadino l'esercitazione dei propri diritti, semplificandogli i mezzi con cui può ottiene i propri dati.

> Per **dato personale** si intende "*qualsiasi informazione relativa a persona fisica identificata o identificabile (**data subject**); una persona identificabile è una persona che può essere identificata, direttamente o indirettamente, in particolare mediante riferimento a un numero di identificazione o a uno o più fattori propri della sua identità fisica, fisiologica, psichica, economica, culturale o sociale*".

Il concetto di dato personale è fondamentale perchè tutto ciò che non rientra nella sua definizione non è coperto legislativamente. Grazie a questo concetto nasce anche quello di processamento dei dati.

> Per **trattamento di dati** si intende "*qualsiasi operazione o insieme di operazioni che è compiuta su dati personali, anche con mezzi automatizzati, come la raccolta, la registrazione, l'organizzazione, la conservazione, l'adattamento o la modifica, il reperimento, la consultazione, l'uso, la divulgazione mediante trasmissione, diffusione o altrimenti messa a disposizione, allineamento o combinazione, blocco, cancellazione o distruzione*".

La responsabilità del rispetto dei principi grava sulle spalle del **titolare del trattamento** (*controller*), ossia la persona fisica o artificiale, l'autorità pubblica, il servizio o qualsiasi altro organismo che, da solo o insieme ad altri, ha determinato le *finalità* e i *mezzi* del trattamento di dati personali.
Il **responsabile del trattamento** (*processor*) è una persona fisica o giuridica, un'autorità pubblica, un'agenzia o qualsiasi altro organismo che tratta dati personali per conto del titolare del trattamento.
Vi sono poi infine gli **addetti al trattamento** che sono le persone che fisicamente trattano i dati e sono nominati dal responsabile.

Il trattamento dei dati personali può avvenire solo quando vengono rispettate:

- **Trasparenza**: l'interessato ha il diritto di essere informato quando è in corso un trattamento dei suoi dati personali. Il titolare del trattamento deve fornire tutte le informazioni necessarie per garantire la correttezza del trattamento;
- **Scopo legittimo**: i dati personali possono essere trattati solo per scopi legittimi e non possono essere trattati per motivi che vanno oltre;
- **Proporzionalità**: se c'è un obiettivo vuol dire che deve essere perseguito utilizzando solo i dati necessari, quindi devono essere usati solo i dati che sono utili al fine dell'obiettivo e non devono essere eccessivi.

**Trasparenza** La persona ha il diritto di:

- **accedere** a tutti i dati personali che lo riguardano;
- richiedere la **rettifica**, la **cancellazione** o il **blocco** dei dati incompleti, inesatti o non trattati nel rispetto delle norme sulla protezione dei dati.

**Scopo Legittimo** I dati devono essere processati solo se uno dei seguenti punti risulta vero:

- la persona ha dato il consenso per il trattamento, che deve essere opzionale e volontario;
- il trattamento è necessario per l'esecuzione di un contratto;
- il trattamento è necessario per adempiere un obbligo legale;

- il trattamento è necessario al fine di tutelare gli interessi vitali dell'interessato;
- il trattamento è necessario per il compito svolto nel pubblico interesse come statistiche per politca, ricerca, ecc...;
- il trattamento è necessario per le finalità dei legittimi interessi perseguiti dal titolare del trattamento, salvo che su tali interessi prevalgano gli interessi per i diritti e le libertà fondamentali dell'interessato.

**Proporzionalità** Al fine di rispettare la proporzionalità del trattamento devono valere tutte le seguenti condizioni:

- i dati devono essere accurati e, ove necessario, aggiornati;
- i dati non devono essere conservati in una forma che consenta l'identificazione delle persone più a lungo del necessario;
- gli Stati membri devono prevedere garanzie adeguate per i dati personali conservati per periodi più lunghi per uso **storico**, **statistico** o **scientifico**;
- devono essere applicate ulteriori restrizioni ai dati **personali sensibili**;
- l'interessato può opporsi in qualsiasi momento al trattamento dei dati personali a fini di marketing diretto;
- una decisione che produce effetti legali o influisce in modo significativo sull'interessato dovrebbe non basarsi esclusivamente sul trattamento automatizzato dei dati;
- una forma di ricorso deve essere fornita quando vengono utilizzati i processi decisionali automatici.

Si parla di **dati personali sensibili** di una persona quando si riferiscono a credenze religiose, opinioni politiche, salute, orientamento sessuale, etnia, appartenenza a organizzazioni passate.

**Autorità supervisore della privacy** Ogni stato membro deve dotarsi di un'**autorità supervisore** della privacy che è il garante dei dati che:

- deve monitorare il livello di protezione dei dati in quello stato membro;
- fornire consulenza al governo in merito a misure e regolamenti amministrativi;
- avviare un procedimento legale in caso di violazione della normativa sulla protezione dei dati.

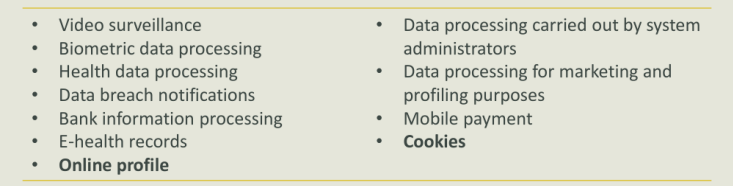
Il titolare del trattamento deve comunicare all'autorità di supervisone le seguenti informazioni:

- il nome e l'indirizzo del titolare del trattamento;
- le finalità del trattamento;
- una descrizione delle categorie dei data subjects e dei dati o delle categorie di dati ad essi correlati;
- i destinatari a cui i dati potrebbero essere comunicati;
- proposta di trasferimenti di dati verso paesi terzi;

- una descrizione generale delle misure adottate per garantire la sicurezza del trattamento.

Queste informazioni sono conservate in un **registro pubblico**.

**Italia** In Italia, la direttiva sulla protezione dei dati è stata implementata dal decreto legislativo n. 196/2003, che conteneva il **Codice italiano sulla protezione dei dati personali**. Questa rimane ancora ad oggi la legge in Italia per la privacy ed è stata leggermente aggiornata/modificata dal GDPR. Inoltre, il *Garante per la Protezione dei Dati Personali* si è impegnato a emanare misure adeguate in merito a molteplici ambiti tra cui (a titolo esemplificativo):




### GDPR

Il **Regolamento Generale sulla Protezione dei Dati** (GDPR) emanato nel 2016 è una legge europea, non più una direttiva, con cui i membri dell'UE intendono rafforzare e unificare la protezione dei dati per tutti gli individui all'interno dell'Unione europea (UE). Gli stati membri a partire dal 2018 sono sotto questo regolamento al di là del fatto che esistano o meno leggi locali nazionali. Gli obiettivi primari del GDPR sono di restituire ai cittadini e residenti il controllo dei propri dati personali e di semplificare il contesto normativo per le imprese internazionali unificando il regolamento all'interno dell'UE. Il GDPR vale ovviamente per cittadini europei risiedenti sul suolo europeo, se uno si sposta in altri paesi è soggetto a trattati internazionali. Presentiamo alcune definizioni riviste dal GDPR.

> **Dati personali**: qualsiasi informazione relativa a una persona fisica identificata o identificabile; una persona fisica identificabile è una persona che può essere identificata, direttamente o indirettamente, in particolare facendo riferimento a un identificatore come un nome, un numero di identificazione, dati sulla posizione, un identificatore online o uno o più fattori specifici della posizione geografica, della identità genetica, mentale, economica, culturale o sociale di quella persona fisica.
>
> **Elaborazione dei dati**: qualsiasi operazione o insieme di operazioni eseguite su dati personali o su insiemi di dati personali, anche con mezzi automatizzati, quali raccolta, registrazione, organizzazione, strutturazione, conservazione, adattamento o alterazione, recupero, consultazione, uso, divulgazione mediante trasmissione, diffusione o

> altrimenti messa a disposizione, allineamento o combinazione, limitazione, cancellazione o distruzione.
>
> **Pseudonimizzazione**: il trattamento dei dati personali in modo tale che i dati personali non possano più essere attribuiti ad una specifica persona interessata senza l'uso di informazioni aggiuntive, a condizione che tali informazioni aggiuntive siano conservate separatamente e siano soggette a misure tecniche e organizzative per garantire che i dati personali non siano attribuiti ad una persona fisica identificata o identificabile.

A livello pratico di database questo si traduce nell'utilizzo di tuple con id numerici casuali per identificare persone in maniera anonima e tabelle esterne che mappano id anonimi con identificativi reali. Si va a separare informazioni personali da chiavi identificative, quest'ultime vengono solitamente anche cifrate.

> **Violazione dei dati personali**: una violazione (accidentale o malevola) della sicurezza che porta alla distruzione, perdita, alterazione, divulgazione o accesso non autorizzati ai dati personali trasmessi, archiviati o altrimenti trattati.

Il GDPR elenca i diritti della persona i cui dati personali vengono elaborati, offrendo alle persone un maggiore controllo sui propri dati personali attraverso:

- la necessità del **consenso chiaro** ed esplicito dell'individuo al trattamento dei dati personali. Questo vale anche per i minori, nel loro caso l'informativa deve essere spiegata in maniera comprensibile con informative semplificate;
- un **accesso più facile** da parte del soggetto ai suoi dati personali;
- i diritti di rettifica, cancellazione e **oblio**;
- il diritto di **opposizione**, compreso l'uso dei dati personali a fini di "profilazione";
- il diritto alla **portabilità dei dati** da un fornitore di servizi ad un altro.

**Privacy by Design and by Default** Il GDPR incoraggia l'utilizzo del principio del Privacy by design e by default che richiede che la protezione dei dati sia concepita nello sviluppo dei processi aziendali per prodotti e servizi. II setting della privacy deve essere impostato al massimo livello per default. Il responsabile del trattamento dovrebbe prendere misure tecniche e procedurali al fine di garantire che l'elaborazione, durante l'intero ciclo di vita, sia conforme al regolamento, inoltre dovrebbero implementare meccanismi per garantire che i dati personali siano trattati solo quando necessario per ogni scopo specifico.

**Privacy by Design** La Privacy by Design (Ann Cavoukian, 2012, ricercatrice canadese) è un approccio all'ingegneria dei sistemi che tiene conto della privacy durante l'intero processo. E' basato su 7 principi fondamentali:

1. proattivo non reattivo, preventivo non correttivo, prevedere delle misure di protezioni del dato che prevengano degli abusi sul dato;

2. privacy come impostazione predefinita;
3. privacy tenuta in conto già nella fase di progettazione dell'intero servizio;
4. funzionalità completa - somma positiva, non somma zero, ovvero, devo rispettare la privacy senza però che ci sia un impendimento per il servizio stesso. Se questo non dovesse accadere allora ci sono solo due casi: 1) servizio mal progettato, 2) servizio che lede palesemente i diritti sulla privacy;
5. sicurezza end-to-end - protezione completa del ciclo di vita;
6. visibilità e trasparenza - mantenerla aperta;
7. rispetto della privacy dell'utente - mantenerlo incentrato sull'utente.

**Pseudonimizzazione** Il GDPR si riferisce alla pseudonimizzazione come un processo che trasforma i dati personali in modo tale che i dati risultanti non possano essere attribuiti ad un soggetto specifico senza l'uso di informazioni aggiuntive, ad esempio crittografando i dati localmente, mantenendo le chiavi per decifrare separatamente dai dati crittografati. Se i dati personali sono pseudonimizzati con adeguate politiche e misure interne, sono considerati effettivamente anonimizzati e non soggetti a controlli e sanzioni del GDPR. Quindi se si garantisce la suddivisione e l'anonimato tra il dato e il riferimento all'individuo di esso, allora non verrà sottoposto a sanzioni tramite GDPR. Il regolamento non riguarda il trattamento di informazioni ritenute anonime, anche a **fini statistici o di ricerca**. Le politiche e le misure che soddisfano i principi della protezione dei dati by design and by default dovrebbero essere considerate adeguate a questo scopo.

**Responsabilità e Responsabilizzazione** Il responsabile del trattamento dei dati deve essere capace di dimostrare in qualsiasi momento che il trattamento viene fatto in compliance con il GDPR e che siano stati implementati i principi di privacy by design e by default. È compito del responsabile del trattamento implementare misure efficaci ed essere in grado di dimostrare la conformità delle attività di trattamento. Questo principio di responsabilità e responsabilizzazione viene realizzato con:

- gli utenti devono essere chiaramente informati dell'**entità della raccolta** dei dati, della **base giuridica** per il trattamento dei dati personali, per **quanto tempo** i dati vengono conservati, se i dati vengono **trasferiti a terzi** e/o al di fuori dell'UE e la **divulgazione** di qualsiasi processo decisionale che viene preso su una base esclusivamente algoritmica;
- gli utenti devono ricevere i dettagli di contatto del responsabile del trattamento e del responsabile della protezione dei dati designato (DPO), ove applicabile;
- gli utenti devono anche essere informati dei loro diritti alla privacy ai sensi del GDPR;
- le **valutazioni d'impatto sulla protezione dei dati** (DPIA) devono essere condotte quando si verificano rischi specifici per i diritti e le libertà degli interessati;

- la valutazione e la mitigazione del rischio e l'approvazione preventiva delle autorità nazionali per la protezione dei dati (DPA) sono necessarie in caso in cui i rischi possano essere elevati;
- devono essere conservati **registri delle attività di elaborazione** che includano finalità del trattamento, categorie coinvolte e termini previsti;
- le registrazioni devono essere rese disponibili all'autorità di controllo su richiesta.

**Responsabile della protezione dei dati** Il GDPR stabilisce la figura del Responsabile della protezione dei dati (DPO), una persona con conoscenza esperta della legge e delle pratiche di protezione dei dati che dovrebbe aiutare il responsabile del trattamento dei dati a monitorare la conformità interna al presente regolamento.
Il DPO è una figura altamente specializzata che deve avere competenze sia di cybersecurity sia giuridiche. Generalmente si parla non di una sola persona, ma di un team tra specializzati di sicurezza e di legge. Ci si aspetta inoltre che il responsabile della protezione dei dati sia competente nella gestione dei processi IT, nella sicurezza dei dati (compresa la gestione degli attacchi informatici) e in altri problemi critici di continuità aziendale relativi alla conservazione e al trattamento di dati personali e sensibili. Le autorità pubbliche e le imprese le cui attività principali sono incentrate sul trattamento regolare o sistematico di dati personali, devono assumere un DPO.

**Violazione dei dati** Il titolare del trattamento ha l'obbligo legale di informare l'autorità di controllo senza indebito ritardo, a meno che sia improbabile che la violazione comporti un rischio per i diritti e le libertà delle persone. Vi sono al massimo 72 ore dopo essere venuti a conoscenza della violazione dei dati per effettuare la segnalazione. Gli individui devono essere informati se si determina un impatto negativo. Il responsabile del trattamento dovrà informare il titolare del trattamento senza indebito ritardo dopo essere venuto a conoscenza di una violazione dei dati personali. Tuttavia, la comunicazione agli interessati non è richiesta se il responsabile del trattamento ha implementato adeguate misure di protezione tecniche e organizzative che rendono incomprensibili i dati personali a chiunque non sia autorizzato ad accedervi, come la crittografia.

**Sanzioni** Possono essere usate le seguenti sanzioni:

- un avvertimento per iscritto in caso di prima non conformità intenzionale;
- audit periodici sulla protezione dei dati;
- un'ammontare fino a 10 milioni di euro o fino al 2% del fatturato mondiale annuo dell'esercizio nel caso di un'impresa, in caso di violazione di alcuni obblighi;
- un'ammontare fino a 20 milioni di euro o fino al 4% del fatturato mondiale annuo dell'esercizio nel caso di un'impresa, se si sono verificate gravi violazioni dei principi.

**Regolamentazione transnazionale** La regolamentazione della privacy internazionale tra Europa e USA è stata definita da:

- **Safe Harbor**: adottato prima dell'avvento del GDPR. Fu stato sviluppato tra il 1998 e il 2000 al fine di impedire alle organizzazioni private all'interno dell'UE o degli Stati Uniti di divulgare o perdere accidentalmente informazioni personali. Le società statunitensi che archiviano i dati dei clienti dovrebbero autocertificare di aver aderito a 7 principi:

  1. **Avviso**: le persone devono essere informate che i loro dati vengono raccolti e come verranno utilizzati;
  2. **Scelta**: le persone devono avere la possibilità di opporsi alla raccolta e di inoltrare il trasferimento dei dati a terzi;
  3. **Trasferimento**: i trasferimenti di dati a terzi possono avvenire solo ad altre organizzazioni che seguono adeguati principi di protezione dei dati;
  4. **Sicurezza**: devono essere compiuti sforzi ragionevoli per prevenire la perdita delle informazioni raccolte;
  5. **Integrità del dato**: i dati devono essere pertinenti e affidabili per lo scopo per cui sono stati raccolti;
  6. **Accesso**: le persone devono essere in grado di accedere, correggere e cancellare le informazioni detenute su di loro;
  7. **Rinforzo**: ci devono essere mezzi efficaci per far rispettare queste regole;

  Dopo che un cliente si lamentò del fatto che i suoi dati di Facebook non erano sufficientemente protetti, la Corte di giustizia della Commissione europea dichiarò, nell'ottobre 2015, che la decisione Safe Harbor non era valida. La Commissione ha tenuto ulteriori colloqui con le autorità statunitensi su "*un quadro rinnovato e solido per i flussi di dati transatlantici*";

- **EU-US Privacy Shield**: il 2 febbraio 2016 la Commissione europea e gli Stati Uniti hanno concordato di stabilire un nuovo quadro per i flussi di dati transatlantici a fini commerciali che rimase in vigore fino al 2020. Uno dei suoi scopi fu quello di consentire alle aziende statunitensi di ricevere più facilmente dati personali da entità dell'UE ai sensi delle leggi sulla privacy dell'UE intese a proteggere i cittadini dell'Unione europea. Nel 2020 il presidente degli Stati Uniti Donald Trump firmò un ordine esecutivo intitolato "*Miglioramento della sicurezza pubblica*" in cui si afferma che la protezione della privacy degli Stati Uniti non sarà estesa al di là dei cittadini o dei residenti statunitensi. Nel luglio 2020 lo scudo UE-USA per la privacy è stato annullato dalla Corte di giustizia europea in quanto non forniva protezioni adeguate ai cittadini dell'UE. I trasferimenti sulla base di questo quadro giuridico sono dichiarati illegali.

# Privacy nell'era dei BIG DATA

## Introduzione

Nel 1996, con l'avvento di Internet, il 24% della popolazione americana aveva dichiarato di aver sperimentato un'invasione della loro privacy. Col passare del tempo tali percentuali sono ovviamente aumentate per colpa sia della diffusione di internet a livello mondiale sia della continua digitalizzazione di strumenti fino a poco fa analogici. Questo è dimostrabile con l'avvento di numerosi scandali di scala globale tra cui Cambridge Analytica, grazie ai quali le persone hanno dato sempre più importanza alla privacy.

*Si può parlare di un compromesso tra convenienza e privacy? Si può disporre di determinati servizi come geolocalizzazione, raccomandazioni su amazon ecc, mantenendo alta la compliance con la privacy? Quali sono i processi informatici che lo permettono?*

Il principio di Privacy by design and default afferma ad esempio che un servizio non deve avere somma 0, cioè non deve essere un servizio inefficiente per preservare la privacy. Ormai la privacy viene riconosciuta come un diritto molto importante anche nelle leggi costituzioni e trattati internazionali.

*Cosa intendiamo per Privacy?*

Non intendiamo solo il concetto di nascondere informazioni, ma in maniera più specifica il possedere i dati che ci riguardano, in questo modo siamo in grado di controllare quanto di noi stessi può essere rilevato agli altri (**Self-Possession**). Tutto questo riguarda anche l'autonomia (**Autonomy**) di pensiero, di movimento e l'integrità delle informazioni personali: se le nostre informazioni raggiungono le altre persone in maniera poco integra si può fornire a queste ultime una visione non corretta di noi stessi che può influenzare le decisioni che potranno essere prese nei nostri confronti in futuro (es. concessione di un prestito). La privacy è quindi collegata ai Diritti Umani: la privacy deve essere garantita sempre in quanto, se viene a mancare, può portare alla discriminazione.

# Era dei Big Data

Una delle caratteristiche principali della nostra epoca è l'abbondanza di dati personali disponibili come: registri di carte di credito, telefonate, mail, account social... Con il termine big data ci riferiamo quindi all'acquisizione ed analisi di grandi collezioni di informazioni, talmente grandi che talvolta non si riescono neanche ad analizzare: questo può essere un vantaggio, ma in contrapposizione abbiamo che non si sa neanche come proteggerli. Alcune problematiche riguardo ad essi sono:

- **Datafication**: tendenza tecnologica a trasformare sempre più aspetti della nostra vita privata e personale in dati che vengono successivamente elaborati come nuova forma di valore;
- **Dati enormi non strutturati e difficili da analizzare**: avendo dati sporchi non sappiamo a che cosa si riferiscono, a volte non si sa nemmeno come interpretarli, ma se non sono interpretabili è difficile sia da rendere utili sia da proteggere;
- **Pattern**: attraverso queste grandi collezioni di dati si possono scoprire dei pattern che altrimenti passerebbero inosservati. Ad esempio, conoscendo la ricerca di determinate parole riguardo sintomi di eventuali malattie su Google si può inferire con probabilità più o meno elevata un certo problema di salute;
- **Memorizzazione indiscriminata**: sotta la spinta dell'enorme successo di Google e la visione di dato personale come nuova risorsa economica, vengono spesso raccolti dati che non servono per uno scopo dichiarato, senza sapere precisamente cosa farne, ma con l'idea di utilizzarli in futuro;
- **Memorizzazione per tempi indefiniti**: in questo modo si può favorire l'estrazione di pattern interessanti riguardanti diversi aspetti delle persone in questione;
- **Legge delle conseguenze non intenzionali**: una tecnologia nasce per una determinata azione, ma poi nel tempo può essere modificata ed applicata per scopi totalmente diversi, non previsti all'inizio. Ad esempio il telefono inizialmente veniva utilizzato solo per telefonare ed inviare messaggi, mentre adesso lo si può utilizzare anche per effettuare pagamenti o identificazione biometrica.

## Cause dei data breaches

I data breaches non avvengono ovviamente per volontà delle aziende, ma sicuramente per loro colpa. Uno studio ha mostrato quali sono le cause/motivi principali:

- **Credenziali compromesse**: al 19% sono causati da password poco sicure e da ingenuità degli utenti del servizio nell'utilizzo di stesse password dappertutto;
- **Servizi cloud malconfigurati**: al 19% i servizi cloud sono configurati in modo tale da non lasciare fuori per esempio gli utenti da accessi ad

informazioni privati che non dovrebbero vedere;
- **Vulnerabilità del software**: al 16% possono essere vulnerabilità software, spesso però sono note da tempo e mai risolte.

Nei primi due casi è sempre l'essere umano a compiere l'errore, nel terzo caso può essere in parte attribuito al software. Per aziende italiane, ad oggi (2022), il costo per un data breach può arrivare fino a 3 milioni di euro, come riporta IBM.

### Linking Attack

Quando Netflix ha pubblicato un dataset anonimo (sostituendo i nomi degli utenti con stringhe numeriche randomiche) contente i voti dei suoi utenti è stato possibile risalire agli utenti stessi collegando le loro recensioni su Netflix con quelle pubbliche rilasciate dagli stessi su IMDB. Questo è stato un problema in quanto vi erano utenti che avevano recensito film per adulti su Netflix, consci del fatto di essere protetti dall'anonimato, ma attraverso questo processo di re-identificazione quelle valutazioni sono state ri-associate al corrispettivo utente. Quindi l'anonimizzazione non può solo consistere nella sostituzione del nome perchè andando ad incrociare i dati da fonti diverse si possono trovare molti più dati di quanti se ne immagini, anche perché le abitudini/gusti sono molto facili da predire attraverso tecniche di AI.

# Privacy nei Sistemi Informativi

## Introduzione

I **sistemi informativi** sono sistemi organizzativi formali e socio-tecnici progettati per raccogliere, elaborare, archiviare e distribuire (anche comunicare verso l'esterno o tra i vari reparti dell'organizzazione) informazioni utilizzate da un'organizzazione.

I **sistemi informatici** sono la parte automatizzata del sistema informativo, nello specifico è un sistema composto da persone e computer che processano o interpretano informazioni. Componenti software e hardware lavorano insieme per acquisire, memorizzare ed elaborare informazioni con lo scopo di:

- **automatizzare** delle attività;
- **supportare le decisioni** prese dai vertici dell'organizzazione. In questo caso le attività di supporto sono analisi, controllo, coordinazione, visualizzazione e reporting.

I sistemi informativi lavorano grazie allo scambio di dati/informazioni: questa operazione di scambio può essere modellata come un set di processi interconnessi dove l'output di uno è l'input di un altro. I dati prodotti appartengono a diverse entità come impiegati, clienti, utenti, fornitori, ecc... e possono essere di diverso tipo: *dati personali*, *email*, *contenuti generati dall'utente*, ecc...
Questi dati scambiati possono essere pubblici e privati (in alcuni casi anche sensibili).

## Come rafforzare la privacy?

Dato che come affermato in Introduzione alla privacy privacy significa anche sicurezza, questa può essere applicata a diversi livelli:

- Educare dipendenti e manager in quanto principali attori del sistema;

- Rafforzare le misure di sicurezza a basso livello: impiegare cifratura e altre tecniche protettive;
- Proteggere le reti e le infrastrutture dei server: uso di firewall;
- Limitare l'accesso fisico alle aree sensibili;
- Usare VPN per il lavoro remoto;
- Tenere aggiornato il software: si evita di lasciare vulnerabilità presenti in versioni precedenti.

# Data Protection e compliance con GDPR

Un sistema informativo conforme al GDPR deve assicurare:

1. **Autorizzazione basata sugli attributi**: un meccanismo di autorizzazione dichiarativo e completamente tracciabile per l'accesso a tutti i dati, in circostanze predefinite;
2. **Anonimizzazione/pseudonimizzazione dei dati**: meccanismi sicuri per la pseudonimizzazione o anonimizzazione delle informazioni degli archivi;
3. **Tracciabilità**: tenere un registro di chi ha creato, modificato o cancellato informazioni, quando e per quale scopo;
4. **Cancellazione dei dati**: meccanismi di sicurezza per garantire il "diritto all'oblio" senza compromettere l'affidabilità del sistema.

## Tecniche di controllo degli accessi per l'autorizzazione

Il **controllo degli accessi** riguarda il limitare l'accesso per le risorse informatiche, soprattutto nei contesti multi-utente con dati condivisi. I requisiti di privacy e sicurezza da garantire nei sistemi informativi insieme aumentano la difficoltà nel realizzare schemi di controllo degli accessi efficienti. Sistemi molto complessi richiederanno sicuramente un controllo degli accessi molto elaborato.

### Role Based Access Control

Messo a punto dal NIST nel 1992 valido per aziende di piccole dimensioni (max 500 impiegati). L'idea alla base è quella di usare i ruoli degli utenti all'interno dei processi del sistema piuttosto che l'ID, permettendo un livello maggiore di astrazione dato che il ruolo agisce come una raccolta di autorizzazioni o diritti. Viene quindi effettuata una prima mappatura tra utente e ruolo e poi una seconda tra ruoli e risorse. L'accesso può essere fornito in lettura o scrittura. In sostanza gli amministratori assegnano il permesso di accesso in base al ruolo, i quali vengono assegnati ai singoli utenti (un utente può avere più ruoli). Gli amministratori che assegnano i ruoli possono anche aggiornare i ruoli o accedere alle autorizzazioni assegnando gli utenti (o rimuovendo gli utenti da) ai ruoli appropriati. Se l'azienda assume un dipendente basta associargli il ruolo e avrà direttamente tutti i permessi.

**Limiti del RBAC** RBAC fu stato dell'arte fino agli anni ′00, soppiantato per i suoi limiti:

- fornisce configurazioni di controllo degli accessi a grana grossa, predefinite e statiche;
- non riesce a fornire un meccanismo flessibile attraverso il quale clienti, utenti e le istituzioni possano esprimere le loro esigenze: non potrebbe rimanere al passo con un'evoluzione del sistema molto veloce con promozioni, linceziamenti o cambiamenti e non permette ad una persona di accedere ai dati solamente per tempo limitato;
- non riesce a catturare lo scopo per il quale i dati verranno rilasciati ai vari stakeholder.

Oggi, la definizione di politiche di autorizzazione basate esclusivamente sul ruolo di un utente non sono abbastanza adeguate. Anche il contesto che circonda tale utente, i suoi dati e l'interazione sono importanti per fornire accesso all'utente corretto, al momento giusto, nella corretta posizione geografica e soprattutto soddisfacendo le regole della normativa.

### Attribute Based Access Control

Si mantiene il concetto di ruolo, ma viene circondato da informazioni sul contesto ugualmente importanti nella decisione di permettere o negare un accesso, nello specifico si tiene in conto di:

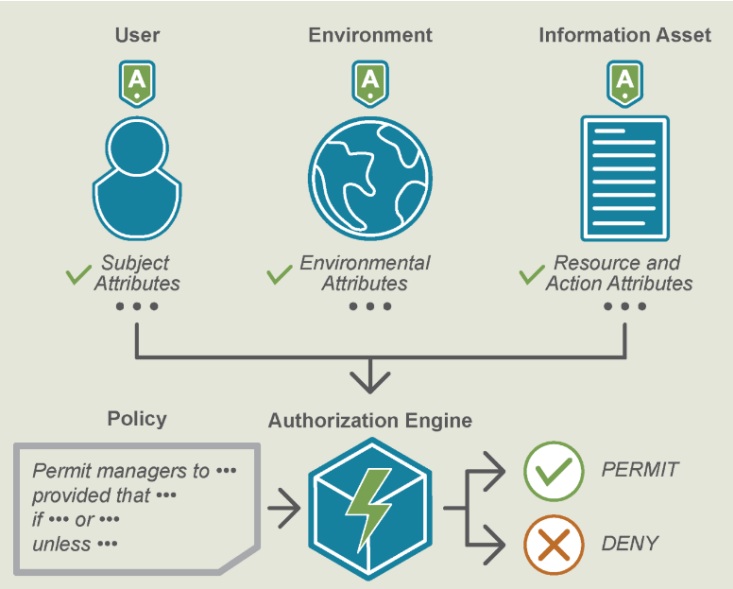
- ruolo dell'utente;
- per chi o cosa l'utente si è collegato;
- a cosa l'utente ha bisogno di accedere;
- da dove l'utente ha bisogno di accedere;
- quando quell'utente ha bisogno di accedere;
- come l'utente accede a tali informazioni.

Conoscere il ruolo di un utente non è più sufficiente per garantire un controllo dell'accesso sicuro e protetto; sono richiesti il contesto e le informazioni sulla relazione tra le varie entità coinvolte nel sistema. ABAC consente ad un'azienda di estendere i ruoli esistenti utilizzando attributi e politiche aziendali. Inoltre, utilizza politiche basate sugli attributi individuali espresse in linguaggio naturale. Avere policies facili da capire insieme al fatto che viene tenuto conto del contesto attorno ad un utente e ciò a cui dovrebbe avere accesso, rende il controllo degli accessi molto più robusto. Nonostante sia simile a RBAC nel senso che adotta anche esso un approccio basato su policies toglie la necessità di essere registrati nel sistema per poter accedere a risorse condivise.

**Come funziona ABAC** Le 3 componenti principali sono:

1. **Utente**: colui che fa l'acesso ad un determinato asset. Gli attributi definibili sul soggetto sono il ruolo/i, gruppi di appartenenza, dipartimento/azienda, livello manageriale, certificazioni/competenze in possesso,

user ID...;
2. **Information asset**: gli oggetti verso cui l'utente fa richiesta di azione del tipo read o write. L'azione può essere arricchita da attributi, e anche le risorse possono avere metadati o tag di documenti;
3. **Ambiente**: identifica il contesto in cui l'accesso è stato richiesto. I suoi attributi sono l'ora e la posizione geografica, il canale di comunicazione, la robustezza dei protocolli di crittografia.

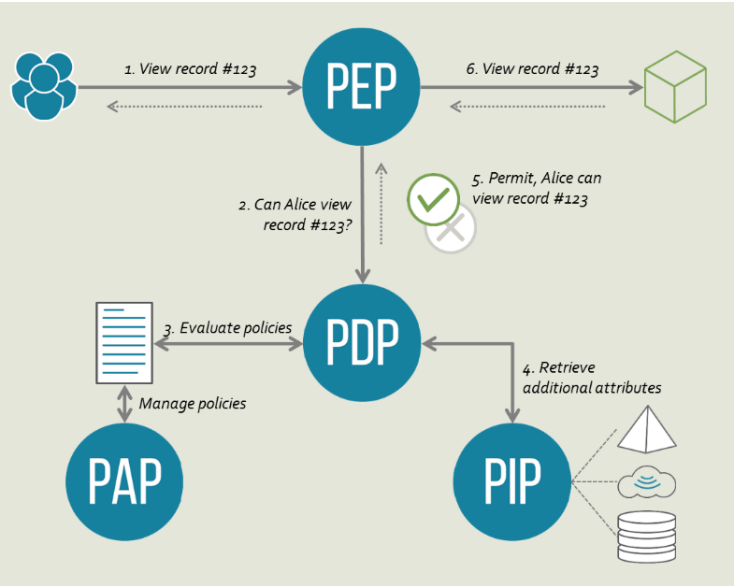



Queste caratteristiche vengono processate dall'**Authorization Engine** che confronta le richieste che sono state effettuate con le policies di cui è dotato e verifica che vengano soddisfatte: in base alla policy e alla richiesta decide se consentire o negare l'accesso alle risorse.

**Authorization Engine** E' composto di 4 diversi moduli che implementano ciascuno diverse funzionalià del motore:

1. **Policy Enforcement Point**: punto in cui arriva la richiesta dall'utente attraverso interfaccia web, mobile o altro e viene subito inoltrata al PDP;
2. **Policy Decision Point**: modulo che prende la decisione riguardo la richiesta usando policies applicabili e informazioni sull'utente e ambiente. La sua decisione viene restituita al PEP che a sua volta la comunicherà all'utente. Si avvale dei moduli PIP e PAP;
3. **Policy Information Point**: componente che memorizza tutti gli attributi che riguardano risorse, soggetti, canali comunicativi ecc... Gli amministratori possono modificare le informazioni direttamente grazie all'aggiunta di nuove policies;

4. **Policy Administration Point**: al suo interno sono memorizzate tutte le policies. Gli utenti di sistema come amministratori hanno entry point privilegiato e possono comunicare direttamente con esso per gestire le policies.




XACML stantard di ABAC (dialetto di XML) è un linguaggio che viene usato per definire politiche di accesso, richieste di accesso e messaggi di interazione tra i diversi moduli del motore. I motori più recenti usano JSON che permette più libertà nella definizione delle policies.

Ogni politica fa parte di un **policy set** e si compone di un target: *soggetto, oggetto e azioni*. Se una richiesta di accesso non matcha con il target nessuna politica del set potrà essere applicata. Ogni politica del set può a sua volta avere un target che rende più specifico il controllo da superare. Il target della politica è seguito dalla regola che a sua volta può definire delle condizioni con funzioni. In questo sistema non solo le organizzazioni, ma anche gli utenti possono definire le proprie politiche che devono essere prese in considerazione. Se ci sono più policies applicabili in un set che vanno in contrasto l'una con l'altra (permit e deny) si usa il **policy combining algorithm** che se è ad esempio specificato con modalità `permit_ovverides` in cui una sola permit ha la meglio su tutti gli altri possibili denies.

## Pseudononimizzazione

La pseudononimizzazione è una procedura di gestione e de-identificazione dei dati mediante la quale i campi identificativi della persona all'interno di un record vengono sostituiti da uno o più identificatori artificiali o pseudonimi. Un singolo

pseudonimo sostituito per ogni campo sensibile rende il record di dati meno identificabile rimanendo però sempre idoneo all'analisi/elaborazione dei dati. Questa procedura ci permette quindi di *scollegare* il dato dal data subject. A livello pratico di database questo prevede di memorizzare la chiave che permette la re-identificazione separatamente rispetto al dato. La parola **pseudo** si riferisce proprio alla possibilità di riottenere l'identificazione iniziale, se così non fosse si parlerebbe di anonimizzazione vera e propria. La differenza è quindi che con un dato anonimo, ottenuto con pseudononimizzazione si può ritornare al dato originale grazie a procedure di re-introduzione dei dati. Mentre i dati anonimi non possono mai essere ripristinati al loro stato originale. Esistono due approcci noti per la pseudononimizzazione.

### Column Encryption

I dati sensibili o che possono portare all'identificazione devono essere cifrati nel server. E' una funzionalità progettata per proteggere i database in cui sono archiviati dati sensibili, consentendo ai client di crittografare i dati sensibili all'interno delle applicazioni client e di non rivelare mai le chiavi di crittografia al motore di database. Di conseguenza, la crittografia fornisce una separazione tra coloro che possiedono i dati (e possono visualizzarli) e coloro che gestiscono i dati (ma non dovrebbero avere accesso). Se il database non può decifrare a sua volta i dati vuol dire che quei dati sono veramente al sicuro, poiché in seguito ad un attacco e non troveranno le chiavi utili per decifrare i dati. Quando si configura la **column encryption**, è possibile specificare le informazioni sull'algoritmo di crittografia e le chiavi crittografiche utilizzate per proteggere i dati nella colonna. Esistono 2 tipi di chiavi:

1. **Column encryption keys**: utilizzate per crittografare i dati in una colonna (una chiave per colonna);
2. **Column masters keys**: una chiave master permette di cifrare una o piu colonne.

Il DBMS archivia la configurazione della crittografia per ogni colonna nei metadati del database (le chiavi non vengono mai archiviate o utilizzate in un testo normale degli scambi). Il DBMS memorizza i valori crittografati della colonne e le informazioni sulla posizione delle chiavi master, che sono archiviate in archivi di chiavi attendibili esterni. Il DBMS supporta alcune query sui dati cifrati, a seconda del tipo di crittografia per la colonna:

- **Deterministica**: preso un certo valore questo viene cifrato sempre allo stesso modo, consentendo quindi di usarlo come chiave per fare *join*, *group by*, indicizzare le colonne e raggruppare i dati stessi. Potrebbe però consentire a utenti non autorizzati, con l'uso della brute force, di trovare i valori crittografati esaminando pattern che si ripetono nella colonna, specialmente se l'insieme dei valori è molto piccolo, come True / False o regione Nord / Sud / Est / Ovest;
- **Randomizzata**: oltre a cifrare i dati utilizzando una chiave, per cifrare

utilizza seed/timestamp o informazioni che rendono la cifratura meno prevedibile. La crittografia non deterministica è più sicura, ma impedisce la ricerca, il raggruppamento, l'indicizzazione e l'unione(join) su colonne crittografate.

Generalmente si usa la deterministica per informazioni private, ma non identificabili di una persona come il comune di nascita, mentre la randomizzata per informazioni molto personali come il codice fiscale.

### Dynamical Data Masking

Limita l'esposizione dei dati sensibili a persone che non hanno tale diritto, ma a differenza della tecnica precedente le informazioni sono in chiaro e vengono mascherate solamente per l'invio al client. Evita l'estrazione dei dati offuscandoli, questa funzione è già incorporata in MS SQL, ma può essere facilmente implementata in qualsiasi DBMS. Ad esempio, il numero della carta di credito può diventare "xxxx-xxxx-xxx-x815" lasciando intendere che sia una carta di credito ma nascondendone la maggior parte. Alcune possibili funzioni di data masking sono:

- **Default**: in base al tipo di dato lo sostituisce con il valore di default prototipico;
- **Email**: lascia il primo carattere, la @ e il .com poi inserisce tutte x;
- **Random**: sostituisce i valori con altri caratteri indicati nella funzione;
- **Custom**: si lasciano solo la prima e l'ultima lettera e un padding.

Anche se raro può capitare che il mascheramento applicato coincida con il dato reale, questo è un problema da tenere in considerazione. Abbiamo meno privacy rispetto al metodo precedente in quanto i curatori del db possono vedere i dati non mascherati. Esistono inoltre dei privilegi (MASK/UNMASK) che possono essere concessi a una persona o a gruppi per poter vedere i dati non mascherati.

**Limiti del data masking** I limiti del data masking sono:

- è possibile bypassare il mascheramento con metodi di inferenza o forza bruta;
- è progettato per semplificare lo sviluppo dell'applicazione limitando l'esposizione dei dati in una serie di query predefinite utilizzate dall'applicazione;
- utenti non amministratori con query ad hoc e le giuste autorizzazioni possono applicare tecniche per ottenere l'accesso ai dati effettivi. Se è necessario concedere tale accesso ad hoc, bisogna revisionare e monitorare tutte le attività del database per mitigare questo scenario.

Il Dynamic Data Masking non dovrebbe essere usato come misura isolata per proteggere completamente i dati sensibili dagli utenti che eseguono query ad hoc sul database. È appropriato per prevenire l'esposizione accidentale di dati sensibili, ma non proteggerà da intenti maliziosi di inferire i dati sottostanti. È

importante gestire correttamente le autorizzazioni sul database e seguire sempre il principio delle **autorizzazioni minime richieste**. È importante abilitare l'**Auditing** per tenere traccia di tutte le attività che si svolgono sul database.

## Auditing

L'auditing è un processo di **esamima** e **validazione** di documenti, dati, processi, procedure e sistemi. L'auditing permette quindi di implementare il principio di tracciabilità delle operazioni in un sistema informativo secondo la compliance del GDPR. L'auditing può essere effettuato a diversi livelli nel sistema e si parla di **log di audit** come il documento che contiene tutte le attività che vengono verificate con l'auditing in ordine cronologico. Gli obiettivi dell'auting sono: insieme di regole aziendali, controlli di sistema, regolamenti governativi o politiche di sicurezza. Definizioni utili:

- **Auditor**: una persona autorizzata per fare l'auditing;
- **Procedura di audit**: serie di istruzioni per il processo di audit;
- **Rapporto di audit**: un documento che contiene i risultati dell'audit;
- **Audit trial**: registrazione cronologica di modifiche ai documenti, modifiche ai dati, attività di sistema o eventi operazionali;
- **Data audit**: record cronologico delle modifiche ai dati archiviate nel file di registro o in una tabella del database che viene memorizzato e non cancellato fino al prossimo audit;
- **Database auditing**: record cronologico delle attività del database come accessi ed up/down.

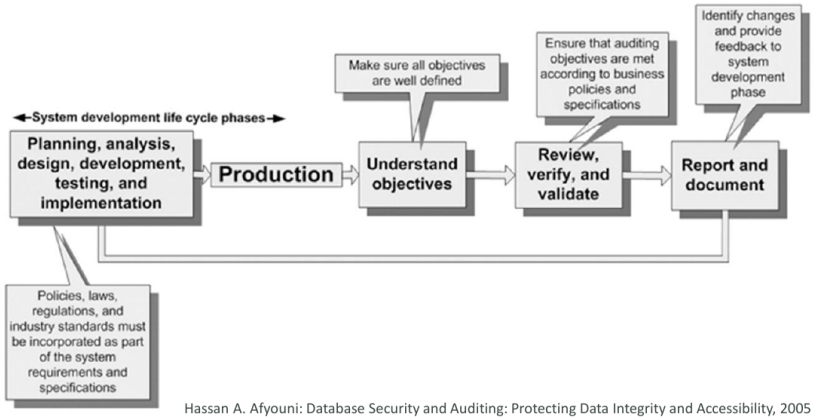
Esistono 2 tipi di auditing:

- **Auditing Interno**: esame delle attività condotte da membri del personale dell'organizzazione sottoposta al processo di auditing. Viene generalmente fatto per preparsi all'auditing esterno;
- **Auditing Esterno**: esame delle attività dell'organizzazione condotte da una terza parte indipendente preposta per tale attività. Esso può dare luogo a delle sanzioni se l'ente esterno verifica che ci sono degli inadempimenti all'interno dell'azienda.

Le **attività di auditing** possono essere le seguenti:

- individuare le problematiche di sicurezza che devono essere affrontate;
- definire piani, politiche e procedure per lo svolgimento degli audit;
- organizzare e condurre gli audit interni;
- assicurare che tutte le voci contrattuali siano soddisfatte dall'organizzazione che si sta verificando;
- agire come collegamento tra la società e il gruppo di controllo esterno;
- fornire consulenza al Dipartimento legale.

Il **processo di auditing** assicura che il sistema funzioni e sia conforme alle politiche, ai regolamenti e alle leggi e viene eseguito dopo la messa in produzione del prodotto:



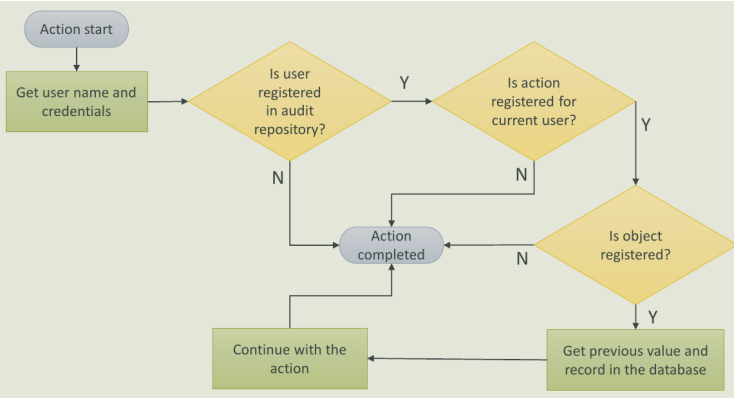

Il processo di auditing non viene fatto con scopo investigativo, ma per aiutare il sistema informativo a risolvere i loro problemi ed essere compliance con il GDPR. Quando il processo di auditing viene fatto su un database gli obiettivi riguardano la verifica di:

- Integrità dei dati;
- Utenti e ruoli dell'applicazione;
- Riservatezza dei dati;
- Controllo degli accessi;
- Modifiche ai dati;
- Modifiche alla struttura dei dati;
- Disponibilità del database o dell'applicazione;
- Controllo dei cambi;
- Relazioni sull'auditing.

Un **modello di auditing** specifica quali sono gli obiettivi dell'auditing e cosa si fa per raggiungerli e quali dati vengono usati per tale processo. Può essere implementato con funzionalità integrate o con il proprio meccanismo, utilizzando PLSQL o altri linguaggi. Le informazioni registrate sono:

- Stato dell'oggetto prima che l'azione fosse intrapresa;
- Descrizione dell'azione che è stata eseguita;
- Nome dell'utente che ha eseguito l'azione.

Un esempio di modello di auditing è il seguente:




L'auditing permette di mettere in atto uno degli aspetti più importanti del GDPR, l'**accountability** poichè si responsabilizzano gli utenti in quanto, sapendo che avverrà un processo di auditing, evitano di fare cavolate.

## Conclusioni

Per rendere Sistemi Informativi e Privacy conformi al GDPR, è necessario prendere in considerazione più misure di sicurezza:

- Controllo degli accessi basato sull'attributo (ABAC);
- Crittografia (crittografia sempre crittografata / crittografia trasparente dei dati);
- Usare Dyamic data masking;
- Implementare l'auditing.

*Ma è tutto questo necessario per proteggere la privacy dei dati memorizzati?*
Sicuramente il livello di protezione è alto, ma comunque non ci garantisce una sicurezza totale.

# Statistical Disclosure Control

## Introduzione

Al giorno d'oggi internet offre opportunità per la raccolta e la condivisione di informazioni sensibili sulla privacy dei suoi utenti sia attraverso dispositivi di cui ognuno è munito come smartphone, smartwatch, pc, ecc... sia mediante applicazioni come i social media. La preoccupazione quindi è quella di mantenere al sicuro questi dati attraverso l'investigazione di aspetti diversi, incluso il problema della protezione delle informazioni rilasciate contro inferenza e linking attack. Bisogna anche tenere in conto che ci sono sempre maggiori norme e leggi che spingono alla pubblicazione di dati open che possono essere usati per migliorare aspetti della società attraverso analisi (dati medici nel periodo covid per velocizzare ricerca al vaccino) con tecniche di:

- machine learning;
- Data mining;
- Statistical inference;
- Big Data analytics (analisi di dati di grossa mole).

Tali dati rilasciati possono essere utilizzati per dedurre informazioni che non erano destinate alla divulgazione (disclosure), bisogna quindi effettuare la pubblicazione dei dati mediante accorgimenti che permettano di anonimizzarli e renderli irriconducibili alla singola persona. In generale una **disclosures** è un'attribuzione impropria di informazioni ad un certo data respondent che sia un individuo o un'organizzazione. Nel primo caso abbiamo un problema di privacy, nel secondo un problema di confidenzialità. La differenza è che alcune informazioni possono essere considerate confidenziali poichè stabilito da un'azienda, mentre altre possono essere private poichè deciso dalla legge. La disclosure può avvenire in diversi modi:

- Grazie ai soli dati rilasciati;
- Risultato della combinazione tra dati rilasciati e informazioni pubbliche;
- Possibile solo attraverso la combinazione di dati rilasciati con altre fonti

dettagliate di dati esterne che potrebbero essere disponibili al pubblico (come i social media che sono ricchi di informazioni personali).

Quando si rilasciano dati, il rischio di divulgazione dei dati rilasciati dovrebbe essere molto ridotto!
Abbiamo due tipi principali di forme in cui vengono pubblicati i dati per motivi di ricerca.

### Macrodati

In passato i dati venivano rilasciati in forma tabellare (macro) attraverso tabelle riassuntive di statistiche su problematiche che riguaradavano la popolazione, si parla di **database statistici**. I dati vengono presentati in maniera aggregata, non abbiamo recordi di singole persone, anche se ci possono essere delle celle a rischio laddove gli aggregati riguardano un numero esiguo di persone e quindi facilmente riconoscibili. Le tabelle di macrodati possono essere classificate nelle seguenti due categorie:

- **Tabelle di Conteggio/Frequenza**: ciascuna porzione della tabella contiene il numero di intervistati (conteggio) o la percentuale di intervistati (frequenza) che hanno gli stessi valori sugli attributi dell'analisi;
- **Dati di tipo Magnitudine**: ogni cella della tabella contiene un valore aggregato di una quantità di interesse su tutti gli attributi di analisi associati alla tabella.

Gli intervistati (respondent) possono essere diversi tipi di attori: aziende, autorità, singole persone, ecc..., da cui vengono raccolti dati e informazioni per la compilazione di statistiche.

### Microdati

Col passare del tempo, per prendere decisioni più precise, si è passati al rilascio di specifici set di dati (micro) i quali sono sia più utili ai fini della ricerca, ma sono anche facilmente soggetti a rischio di privacy. In questo caso ogni riga si riferisce ad uno specifico respondent, mentre nel caso precedente la tabella faceva riferimento ad un gruppo di respondent. Qua il rilascio di tali dati è molto rischioso. I principali accorgimenti da prendere in considerazione sono quelli per proteggere l'identità, attributi ed evitare il rischio di inferenza. Anche rimuovendo gli identificatori nei microdati non è detto che la privacy sia garantita.

## Information Disclosure

Abbiamo tre tipi diversi di information disclosure:

- **Identity Disclosure**: un respondent viene identificato dai dati rilasciati. Si ha quando una parte terza è in grado di identificare un soggetto a causa dei dati rilasciati, anche usando altre informazioni per incrociare i risultati.

Tuttavia, bisogna precisare che rivelare che un individuo è un respondent di una collezione di dati potrebbe o no violare i requisiti di confidenzialità in particolare se abbiamo:
  - **Macrodata**: rivelare l'identità non è generalmente un problema, a meno che l'identificazione non porti a rivelare informazioni riservate (*attribute disclosure*);
  - **Microdata**: l'identificazione è generalmente considerato un problema, poiché i record dei microdati sono molto dettagliati (implica generalmente anche l'*attribute disclosure*);
- **Attribute Disclosure**: con il rilascio di alcuni dati vengono rivelate anche informazioni sensibili riguardo ad un respondent, ed inoltre è possibile attribuire a lui tali informazioni. Può verificarsi sia quando le informazioni riservate vengono rivelate esattamente sia quando possono essere stimate molto accuratamente;
- **Inferential Disclosure**: i dati rilasciati consentono di determinare il valore di alcune caratteristiche di un respondent in modo accurato attraverso deduzioni, cosa che in caso contrario non sarebbe stata possibile. Si verifica quando le informazioni possono essere dedotte con elevata sicurezza dalle proprietà statistiche dei dati rilasciati. È difficile prendere in considerazione questo tipo di divulgazione per due ragioni:
  - Se la divulgazione equivale all'inferenza, non è possibile rilasciare dati;
  - Le inferenze sono progettate per prevedere il comportamento aggregato, non i singoli attributi, e sono quindi spesso predittori inadeguati dei singoli valori dei dati.

Per evitare tutto questo abbiamo 2 tipi di azioni che possiamo fare, che dipendono dal tipo di dato che vogliamo proteggere:

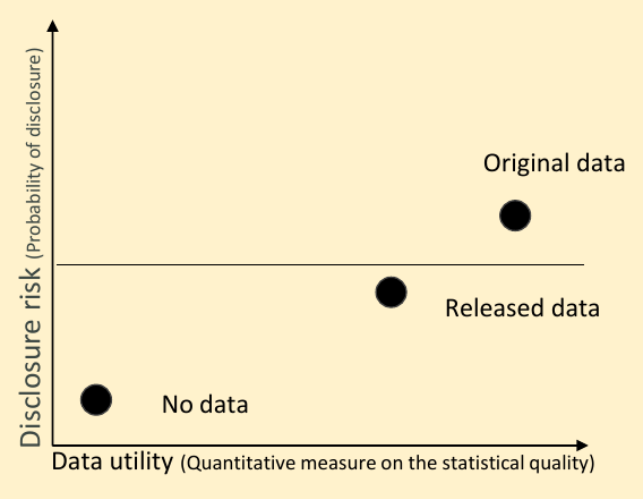
- **Restricted Data**: rimuovere non solo tutte le informazioni identificative (nel caso dei microdati), ma anche qualsiasi altra informazione considerata pericolosa;
- **Restricted Acess**: imporre condizioni di accesso sui dati in cui posso ottenere dati più o meno precisi a seconda del livello di accesso che ho.

Posso usare ovviamente anche una combinazione di entrambe le tecniche.

## Statistical Disclosure Control

La SDC è una collezione di metodi usati come parte del processo di anonimizzazione che, a partire dai dati in chiaro, attraverso tecniche di *pulizia* (manipolazione) permetto di ottenere dati difficili da re-identificare o su cui fare attribute disclosures (molto difficile, ma generalemente non impossibile). La SDC non è specificatamente pensata per la privacy, ma più per garantire la confidenzialità dei dati. Per essere applicati per garantire la privacy questi metodi vanno mappati in maniera diretta/indiretta, perfetta/imperfetta su metodi di protezione dei dati. A livello pratico si deve affrontare un trade-off tra utilità

pratica del dato rilasciato e probabilità di disclosure. Deve essere rilasciato un dato il più utile possibile senza però superare una soglia di probabilità, che è un parametro molto difficile da stabilire:




## Procedura di SDC

Per ogni procedura di Statistical Disclosure Control dobbiamo affrontare 5 steps successivi:

1. **Data**: dobbiamo definire i dati che vogliamo proteggere: tabella di micro/macro data;
2. **Metodo**: applicazione del metodo di SDC andando a scegliere i suoi parametri che determinano anche il livello di protezione garantito;
3. **Utilità**: misurare l'utilità dei dati che abbiamo ottenuto dal passo precedente;
4. **Rischio**: calcolare il rischio di disclosure che si ottiene con il metodo scelto;
5. **Compromesso**: cercare un compromesso tra utilità e rischio con analisi *ex-post*, ossia a valle del processo SDC di modifica (metodi *ex-ante* sono ad esempio la k-Anonymity e la Differential Privacy).

### Metodo di SDC

Un metodo di SDC, oltre ai noti micro/macro dati può essere anche applicato ad un sistema di queries in cui le informazioni non vengono rilasciate tutte insieme in un file, ma in maniera graduale attraverso interfaccia grafica o API di interrogazione.

**Metodi per Macrodati** Per i macrodati abbiamo a disposizione metodi di 2 tipologie diverse:

- **Non-Perturbativi**: non modificano i valori nelle celle, ma sopprimono i valori che sono considerati a rischio. Una delle tecniche è la **cell suppression** che prevede la:
  - **Primary Suppression**: rimozione dei valori considerati a rischio come percentuali o conteggi molto piccoli di cui è facile inferire l'identità delle persone;
  - **Complementary (o Secondary) Suppression**: a volte la primary da sola non basta per garantire la protezione dei dati, ma devo eliminare anche altri valori collegati, che di base non sarebbero a rischio. Ovviamente si cerca sempre di sopprimere la minor quantità possibile di informazione;
- **Perturbativi**: metodi di modifica dei valori delle celle con lo scopo di limitare la possibilità d'inferenza. I più conosciuti sono:
  - **Random Rounding**: i valori numerici nelle celle sono arrotondati in maniera casuale per difetto o per eccesso all'unità, decina o centinaia, ecc... più prossima;
  - **Controlled Rounding**: arrotondamento controllato, uguale al precedente, ma in cui si vogliono mantenere i marginali delle righe e delle colonne;
  - **Controlled Tabular Adjustment**: date delle celle che sono considerate *sensibili*, queste vengono sostituite utilizzando il valore sicuro a loro più prossimo. Tutto questo viene fatto ovviamente in ottica di preservare l'additività/marginali delle tabelle statistiche considerate.

**Metodi per Dati ottenuti con queries** Per i dati che vengono rilasciati tramite queries possiamo applicare 2 possibili tipi di limitazione:

- **Query Perturbation**: si può applicare la perturbazione sul database a monte e poi fare le queries su tale db (*input perturbation*), oppure perturbazione a valle, sul risultato della query con rumore che segue una certa distribuzione (*output perturbation*);
- **Query Restriction**: riguarda la possibilità di poter bloccare alcune interrogazioni e rifiutare alcune queries. Per fare questo devo tener traccia di tutte le precedenti queries dato che l'utente malevolo potrebbe risalire all'informazione cercate mediante una sequenza ad hoc di interrogazioni. Le restrizioni vengono applicate quindi generalmente per queries troppo stringenti che richiedono dati aggregati troppo piccoli.

**Metodi per Microdati** Per i dati in cui ogni tupla/record rappresenta una singola persona possiamo avere:

- **Data Masking**: l'idea è generare delle versioni modificate $X'$ del database originale $X$ che si possono ottenere attraverso un:
  - **mascheramento perturbativo**: le tecniche più note sono:
    - **Noise addition**: aggiunta di vettore di rumore ad ogni tupla, approccio ovviamente applicabile quando i dati sono numerici. Si

può pensare di aggiungere rumore in modo da mantenere medie e correlazioni tra i dati;
    - **Microaggregation**: si vanno ad aggregare tuple con valori molto simili tra loro, con similarità vista come un concetto intra-gruppo. Non è un problema di clustering perchè non tengo conto della dissimilarità tra gruppi diversi. Inoltre si considerano gruppi molto piccoli andando ad aggregare anche solo 2/3 tuple per volta. Aggregando si tengono come valori le medie se i dati di partenza erano numerici o la moda in caso di dati categorici;
    - **Data Swapping**: consiste nell'ordinare i valori di un attributo secondo un certo ordine (ad esempio ascendente in base alla frequenza) per poi swappare i valori di rango simile;
    - **Post Randomization**: lavora principalmente su dati categorici, si cambiano gli attributi secondo una matrice stocastica seguendo generalmente un processo Markoviano;
  - **mascheramento non perturbativo**: le tecniche più note sono:
    - **Sampling**: anzichè pubblicare l'intero dataset ne pubblico un sottoinsieme ovviamente rimuovendo le informazioni identificative;
    - **Generalization**: attributi di tipo categorico vengono generalizzati in valori meno specifici mediante gerarchia di generalizzazione, i valori numerici possono essere rimpiazzati con intervalli;
    - **Top/Bottom Coding**: valori sopra o sotto soglia vengono rimpiazzati corrispondentemente con i valori di margine superiore ed inferiore;
    - **Local Suppression**: soltanto alcuni attributi individuali vengono soppressi e si vanno ad aumentare gli insiemi di record che hanno gli stessi valori su combinazione di attributi;
- **Data Synthesis**: a partire da $X$ genero un dataset completamente sintetico $X'$ che però preservi alcune proprietà di $X$ come media/varianza/ecc... In particolare abbiamo:
  - **Goal**: rilasciare un dataset che mantengano la confidenzialità dei data subjects;
  - **Metodo**: la sintesi potrebbe avvenire anche solo per categorie di persone a rischio, fare una scelta orizzontale (gruppi) di persone da sintetizzare, abbiamo a disposizione due diversi metodi:
    - **Fully Synthetic Data**: si genera un dataset completamente sintetico;
    - **Partially Synthetic Data**: si generano soltanto valori sintetici per gli attributi considerati sensibili;
  - **Problemi**: quando applichiamo la sintesi di tutto o parte del dataset dobbiamo:
    - garantire che i dati sintetici siano ancora utili e permettano analisi statistiche;
    - fare attenzione al fatto che la generazione è strettamente dipendente dal modello scelto;

    * ci potrebbe essere problema di overfitting se il dataset sintetico è troppo simile a quello reale da cui siamo partiti.

#### Utilità del SDC

Per fare il trade-off sopra accennato bisogna calcolare una certa misura di utilità. Il calcolo di questa è molto difficile perchè se rilascio un dataset general purpose per cui non so l'utilizzo che ne verrà fatto e ipotizzo io un certo obiettivo per il calcolo dell'utilità, se poi cambia l'obiettivo di analisi non è detto che il dataset risulti ancora utile. Generalmente si calcola l'**information loss** ossia misure che, usando la teoria dell'informazione, calcolano quanto l'informazione di partenza è stata modificata, con una delle precedenti modalità. Vediamo alcune loss che possono essere usate:

- **Macrodati**: abbiamo misure diverse a seconda della metodologia come:
  - **Cell Suppression**: in questo caso si va a calcolare il numero o la somma dei valori delle celle oggetto di soppressione secondaria questo perchè la primaria è minima ed inevitabile mentre è quella complementaria che è problematica;
  - **Rounding and Tabular Adjustment**: possiamo usare la somma delle distanze tra i valori nelle celle vere e quelle perturbate. Posso anche pesare diversamente le celle in modo da avere valore più realistico;
- **Query**: abbiamo ovviamente anche qui misure diverse a seconda del metodo:
  - **Query Perturbation**: differenza tra il vero valore restituito dalla query e quello ottenuto dalla query perturbata (può essere anche calcolata in termini di media e varianza del rumore aggiunto);
  - **Query Restriction**: il numero delle queries che vengono rifiutate;
- **Microdati**: ci sono misure specifiche in base all'uso dei miei dati, non c'è nè una sempre vera ed applicabile ad esempio:
  - **Statistiche di Base**: media/varianza/correlazioni;
  - **Score**;
  - **Distanza**: tra dati originali e dati rilasciati con misure probabilistiche che misurano la distanza tra distribuzioni di probabilità come la *Jensen-Shannon divergence* e la *Kullback-Leibler divergence*.

## Conclusione

La SDC prevede il calcolo del rischio ed utilità a valle, ex-post, a priori non posso sapere quale sarà la possibilità di re-identificazione dei miei data subject, così come non posso sapere quale sarà l'utilità che otterrò. Se voglio un metodo che opera ex-ante, a monte, userò un **modello di privacy** come la Differential Privacy o la k-Anonymity che dipendono da uno o più parametri e che mi garantiscono un *upper bound* ad un rischio di re-identificazione. Questi metodi rischiano di essere molto perturbativi e vengono generalmente usati solo per

problemi di privacy, non di confidenzialità.

# k-Anonymity

## Introduzione

Diverse agenzie, istituzioni, uffici ed organizzazioni rendono disponibili al pubblico o a terze parti dati (sensibili) che coinvolgono le persone come i **microdati** per scopi statistici e di ricerca. Spesso questo è richiesto e imposto dalla legge. Ad esempio per provare la bontà di nuovi metodi/algoritmi che vengono sviluppati è necessario pubblicare il dataset anonimo per fare in modo che l'esperimento sia riproducibile. La prima cosa da fare è la **sanificazione** del dataset ossia la rimozione di tutte informazioni che possono portare all'identificazione o re-identificazione, identificatori espliciti. Questo però non basta, è ancora possibile fare **linking attack** sui microdati pubblicati.

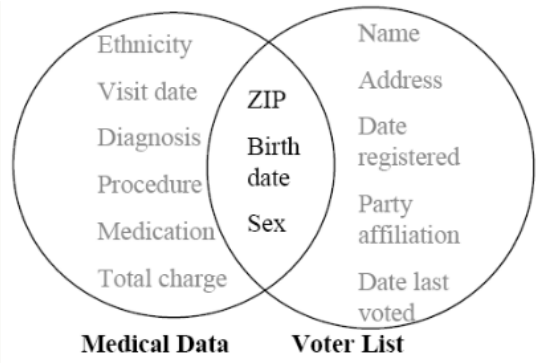
### Linking Attack

Questo attacco è possibile, nonostante la sanificazione del dataset, quando le persone hanno dati disponibili in diverse sorgenti pubbliche. Il linking attack è stato teorizzato dalla ricercatrice Latanya Sweeney nel 2001 che è riuscita a re-identificatore il fascicolo medico del governatore del Massachussetts. Nel MA infatti venivano pubblicati i dati sanitari degli impiegati pubblici in maniera sanificata. Sweeney ha collegato questi dati, pubblicati in maniera pseudo-privata, con la lista pubblica dei votanti grazie ad un join su un set di attributi comuni: `ZIP, birth date, sex`. E' riuscita facilmente a trovare le informazioni sanitarie del governatore grazie al join poichè vi era solo lui, in tutto il MA, con quella combinazione di attributi. Grazie al censimento avvenuto negli USA nel 1990 si è scoperto che l'87% della popolazione americana era unica grazie alla combinazione di questi 3 attributi.

### Tipi di Attributi

Andiamo ad identificare 3 diverse tipologie di attributi nei microdati:

- **Identificatori Espliciti**: rimossi attraverso la sanificazione del dataset;
- **Quasi Identifier**: identificatori come quelli usati da Sweeney, apparentemente innocui, ma che possono essere usati per la re-identificazione delle




persone. Già nel 1986 Dalenius lo definì come attributo che di per sè non è identificativo, ma che in combinazione con altri quasi-identificatori possono creare un identificatore univoco;

- **Attributi Sensibili**: sono i dati che ci interessano per fare analisi statistiche, andando a cercare correlazione tra questi e i quasi-identifiers.

| identifier | quasi identifiers | | | sensitive |
|---|---|---|---|---|
| Name | Birthdate | Sex | Zipcode | Disease |
| Andre | 21/1/79 | male | 53715 | Flu |
| Beth | 10/1/81 | female | 55410 | Hepatitis |
| Carol | 1/10/44 | female | 90210 | Brochitis |
| Dan | 21/2/84 | male | 02174 | Sprained Ankle |
| Ellen | 19/4/72 | female | 02237 | AIDS |

goal of privacy preservation (rough definition)
**de-associate** individuals from **sensitive information**

L'obiettivo degli algoritmi di anonimizzazione è di disassociare l'individuio dall'informazione sensibile.

## $k$-Anonymity

E' stato proposto come modello di privacy da Sweeney e Samarati nel 1998 per risolvere il problema del linking attack, proponendo anche tecniche per produrre datasat k-anonimizzati usando metodi di **generalizzazione** e **soppressione**.

### Definizioni Utili

Dato un dataset $D$ definito su un set di $n$ attributi $\{A_1, A_2, ..., A_n\}$ abbiamo che:

> Un **quasi-identifier** di $D$ è un set di attributi $QI \subset A$ di cui dobbiamo controllare il rilascio. E' quindi l'insieme $QI$ che può portare al

> linking attack. Andiamo a definire con $QI_D = \{QI_1, QI_2, ..., QI_m\}$ l'insieme dei quasi-identifiers per $D$.

Rispetto alla definizione di Dalenius per la Samarati un QI non è un singolo attributo, ma un set. Possiamo quindi dire che:

> $D$ **soddisfa la k-anonymity** sse $\forall QI_i \in QI_D$ e per ogni possibile combinazione dei valori di $QI_i$ che compaiono all'interno di $D$ vi siano almeno $k$ tuple. Ossia se ci sono almeno $k$ occorrenze di qualsiasi combinazione di valori di ogni quasi-identifier di $D$.

Dato che ogni tupla appartiene ad un data subject diverso, quando impostiamo il valore di $k$ stiamo chidendo che ogni persona sia "*mascherata*" da almeno altre $k-1$ con gli stessi valori degli attributi. Implicitamente stiamo quindi dicendo che la probabilità di riuscita di un linking attack è al massimo pari a $\frac{1}{k}$. E' un metodo ex-ante perchè nel momento in cui definisco il valore di $k$ sto definendo la massima probabilità di re-identificazione. La $k$-anonymity richiede quindi che al rilascio della tabella i respondent non siano distinguibili l'uno dall'altro rispetto al set di quasi-identifier. Ovviamente se $k = 1$ la tabella non è anonima per definizione. Bisogna arrivare a produrre tabella $k$-anonima e ci sono 2 tecniche principali per farlo:

- **Generalizzazione**: sostituzione di un valore di un attributo con uno più generale, per farlo devo avere a disposizione una gerarchia di generalizzazione;
- **Soppressione**: tecnica che consiste nel rimuovere singoli valori o più frequentemente un' intera tupla e non è fine a se stessa, ma serve per ridurre il livello di generalizzazione che avrei altrimenti. Ad esempio preferisco togliere un outlier che appiattirlo generalizzandolo.

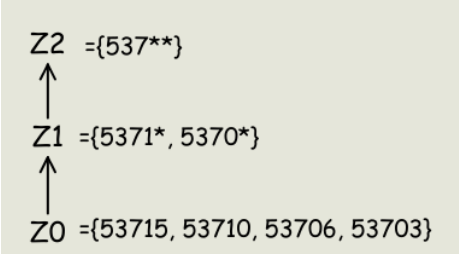
Le due tecniche possono quindi essere in combinazione ed è proprio quello che propone Samarati. Il problema della generalizzazione è che l'eccessiva perturbazione, con appiattimento delle diversità dei dati, può rendere un dataset non più utile per fini statistici. Trovare il corretto $k$ non è semplice ed è frutto di una attenta analisi del rischio che sono disposto ad accettare. Per introdurre i vari algoritmi di $k$-anonymity abbiamo bisogno di una serie di definizioni di base.

### Gerarchia di Generalizzazione del Dominio

Una **gerarchia di generalizzazione del dominio** $DGH_D$ sul dataset $D$ composto di un attributo $A_i$ è un ordinamento parziale sul set dei domini $Dom_{A_i} = \{D_0, D_1, ..., D_n\}$ di $A_i$ che soddisfano le proprietà:

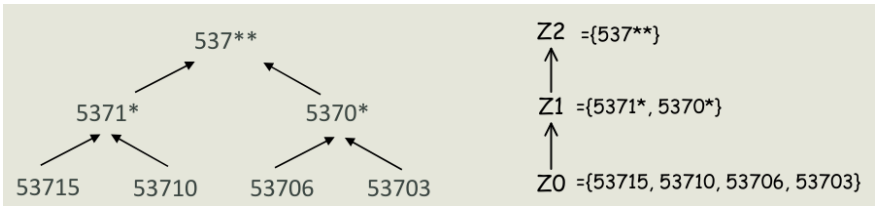
1. ogni $D_i$ è una generalizzazione del dominio di base $D_0$ e ha, all'interno del set, un solo altro dominio $D_j$ che lo generalizza direttamente;
2. ogni dominio massimale del set $Dom_A$ sono dei singletons, ossia contengono un solo valore, per essere sicuri che, nel caso in cui si volesse, ogni attributo

potrebbe essere generalizzato fino ad un solo valore indistinguibile tra tutti i record.




### Gerarchia di Generalizzazione dei Valori

Una **relazione di generalizzazione dei valori** è una relazione che associa ad ogni possibile valore $v_i$ di un dominio $D_i$ un unico valore $v_j$ nel dominio $D_j$ tale che $D_j$ è una generalizzazione diretta del dominio di $D_i$. Definendo questa relazione per ogni dominio $D_i$ di un attributo $A_i$ abbiamo una **gerarchia di generalizzazione dei valori** $VGH_D$ che è un albero/tassonomia dove le foglie sono i valori del dominio di base $D_0$, mentre la radice è il singolo valore del massimale $D_n$.




### Reticolo di Generalizzazione

Andando a generalizzare la definizione e permettendo la presenza di più attributi, abbiamo che data una tupla di domini $DT = \langle D_{A_1}, D_{A_2}, ..., D_{A_n} \rangle$ tale che $D_{A_i} \in Dom_{A_i}$ la **gerarchia di generalizzazione del dominio** di $DT$ è $DGH_{DT} = DGH_{A_1} \times DGH_{A_2} \times \cdots \times DGH_{A_n}$. $DGH_{DT}$ definisce un **reticolo di generalizzazione** i cui elementi sono tuple $DT$ e ogni arco rappresenta un'azione di generalizzazione compiuta su uno degli $n$ attributi. Questo lattice è caratterizzato da avere come elemento infimo la tupla con tutti gli attributi non generalizzati e come elemento supremo la tupla con tutti i domini massimali per ogni attributo.

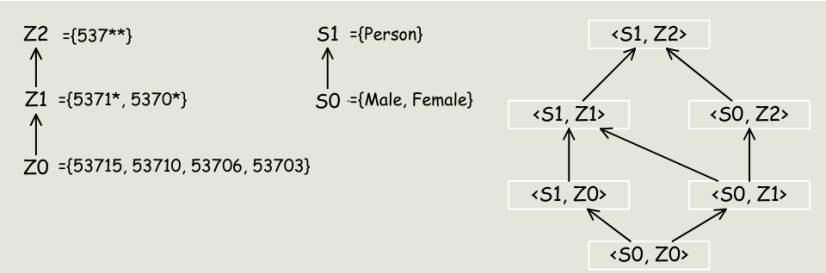



**Tabella Generalizzata con Soppressione**

Dato un set di attributi $A = \{A_1, A_2, ..., A_n\}$ e due tabelle $T_i$ e $T_j$ con entrambe lo schema di attributi $A$. Possiamo dire che la tabella $T_j$ è una **generalizzazione** (*con soppresione di tuple*) della tabella $T_i$ denotata con $T_i \leq T_j$ sse valgono le seguenti due condizioni:

1. il dominio di ogni attributo $A_x$ in $T_j$ è uguale o una generalizzazione del corrispondente in $T_i$;
2. ogni tupla $t_j \in T_j$ ha una corrispondente tupla $t_i \in T_i$ tale che per ogni attributo $A_x$ il valore $t_j[A_x]$ è uguale o una generalizzazione del corrispondente $t_i[A_x]$.

Da questa definizione possiamo notare come in $T_i$ per la condizione 2) ci saranno tutte le tuple di $T_j$, ma il vicersa non è garantito perchè possiamo avere soppressione.

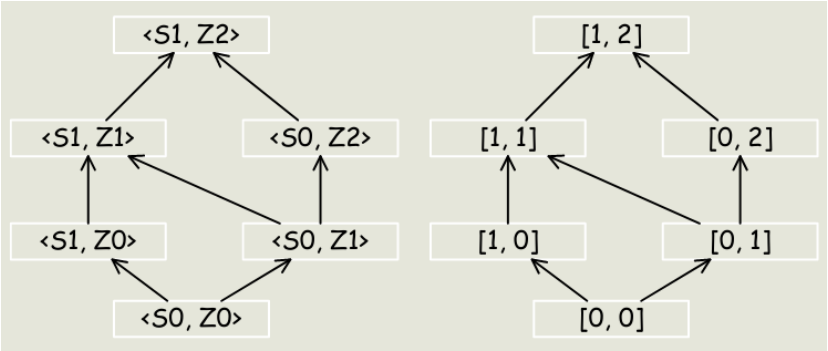
**Vettore di Distanze**

La scelta del valore $k$ è deciso a fronte di un'analisi del rischio, come parametro di design. In particolare se decidiamo che $k$ deve valere almeno un certo valore soglia $k \geq s$ dobbiamo scegliere la generalizzazione che risulta più utili a fini statistici. Date due tabelle $T_i \leq T_j$ per la definizione precedente viene detto **vettore di distanze** da $T_i$ a $T_j$ $DV_{ij} = [d_1, d_2, ..., d_n]$ un vettore di interi dove ogni $d_x$ è la lunghezza del percorso unico tra $dom(A_x, T_i)$ e $dom(A_x, T_j)$ nella gerarchia di generalizzazione del dominio $DGH_{A_x}$:

Questo concetto viene usato nel seguente criterio di generalizzazione.

**Criterio di Generalizzazione** L'idea è la seguente:

1. Costruisci il reticolo di generalizzazione;
2. Trova la generalizzazione minima che soddisfi la $k$-anonymity ossia trova il nodo del reticolo che ha il vettore di distanza dall'infimo con minore norma 2. Questo mi permette di massimizzare l'utilità del dataset che ottengo.




## Algoritmo di Samarati

### Note sull'algoritmo

Facendo uso del criterio precedente l'algoritmo di Samarati risolve un problema più specifico, quello della **generalizzazione $k$-minimale con soppressione**. Date due tabelle $T_i \leq T_j$ e sia $MaxSup$ il parametro che limita il numero massimo di soppressione delle tuple, allora $T_j$ è una generalizzazione $k$-minimale di $T_i$ con soppressione sse:

1. $T_j$ soddisfa la $k$-anonymity e $|T_j| - |T_i| \leq MaxSup$;
2. la soppressione sia minimale tra generalizzazioni con ugual vettore di distanze, ossia:

$$\forall T_x \quad k\text{-}anonima \quad T_i \leq T_x \quad t.c. \quad DV_{ij} = DV_{ix} \quad |T_i| \leq |T_x|$$

3. ogni altra tabella $k$-anonima con più tuple deve essere peggiorante come vettore ossia:

$$\forall T_x \quad k\text{-}anonima \quad T_i \leq T_x \quad t.c. \quad |T_x| \geq |T_j| \Rightarrow DV_{ix} > DV_{ij}$$

Da questa definizione possiamo notare come la condizione sul vettore di distanza prevalga su quella della cardinalità/soppressione. Nonostante le proprietà cercate possono comunque ancora esistere generalizzazioni diverse con ugual norma 2 del vettore di distanza e ugual cardinalità. In questo caso l'algoritmo di Samarati andrà a considerare le generalizzazioni equivalenti anche se una potrebbe essere migliore delle altre. Per arrivare a trovare una generalizzazione $k$-minimale con soppressione della tabella di partenza possiamo implementare diverse procedure a seconda di come applichiamo la generalizzazione e la soppressione:

- **Generalizzazione**: può essere applicata a livello di:
  - **colonna**: la generalizzazione si applica a tutte le tuple per una colonna;
  - **cella**: quando la generalizzazione riguarda una cella o gruppi di celle la stessa colonna può essere generalizzata in maniera diversa in gruppi di tuple, permettendo una grana più fine, mantenendo più informazioni rispetto alla generalizzazione livello colonna;

- **Soppressione**: può essere applicata a livello di:
  - **riga**: tutta la tupla viene tolta;
  - **colonna**: tutta la colonna viene eliminata;
  - **cella**: solo alcuni valori in specifiche celle vengono eliminati.

| Birthdate | Sex | Zipcode |
|---|---|---|
| 21/1/79 | male | 53715 |
| 10/1/79 | female | 55410 |
| 1/10/44 | female | 90210 |
| 21/2/83 | male | 02274 |
| 19/4/82 | male | 02237 |

original microdata

| | Birthdate | Sex | Zipcode |
|---|---|---|---|
| group 1 | */1/79 | person | 5**** |
| | */1/79 | person | 5**** |
| suppressed | 1/10/44 | female | 90210 |
| group 2 | */*/8* | male | 022** |
| | */*/8* | male | 022** |

2-anonymous data

Tutte le possibili combinazioni di questi approcci portano alla definizione di diversi algoritmi. Applicare le procedure a grana più fine, livello di cella, garantisce una tabella finale con maggior contenuto informativo possibile, ma richiede una maggior complessità computazionale. Il problema di trovare la minima tabella $k$-minimale con soppressione delle tuple e generalizzazione degli attributi (colonne) è **NP-hard**, con tempo esponenziale nel numero di attributi che compongono i quasi-identifiers, vogliamo quindi usare delle euristiche che ci permettano di trovare delle soluzioni subottimali ma che terminino in un tempo accettabile. L'approccio esaustivo di ricerca risulta fattibile solo in contesti in cui abbiamo un numero molto piccolo di colonne.

### Pseudocodice

L'algoritmo di Samarati usa le seguenti considerazioni:

1. ogni percorso nel reticolo di generalizzazione è una **strategia di generalizzazione**, l'algoritmo cerca la generalizzazione che è **localmente minima** ossia vuole trovare il nodo più basso che soddisfa la $k$-anonymity in ogni percorso;
2. ogni generalizzazione globalmente $k$-minimale con soppressione è ovviamente anche localmente minimale rispetto ad un percorso, il contrario non è vero. Bisogna fare trade-off tra salire nel reticolo che garantisce meno soppressione di tuple, ma maggior generalizzazione e il viceversa;
3. sfrutta il fatto che se non esiste una soluzione $k$-minima ad un livello $h$ con numero di soppressione accettabile ($\leq MaxSup$) allora non ce ne saranno manco ad un livello più basso. Questa proprietà è molto utile nella ricerca.

L'algoritmo sfrutta quanto detto facendo una **binary search** nel reticolo dei vettori delle distanze nel seguente modo, dato $h$ l'altezza massima della porzione in cui ci troviamo:

1. calcolo le soluzioni all'altezza $\lfloor h/2 \rfloor$;
2. se c'è soluzione che soddisfa la $k$-anonymity con soppressione, provo a scendere un po' di più migliorando l'utilità del dataset riducendo la gen-

eralizzazione e quindi ricorsivamente calcolo tutte le soluzioni all'altezza $\lfloor h/4 \rfloor$;
3. se non c'è soluzione salgo nel reticolo il minimo possibile e calcolo tutte le soluzioni all'altezza $\lfloor 3h/4 \rfloor$;
4. finchè non viene trovata la soluzione ottimale i passi vengono ripetuti in maniera ricorsiva aggiornando ovviamente il valore di $h$ in modo da esplorare la corretta porzione dello spazio.

Per ridurre la complessità adotta una **matrice di distanza dei vettori** in modo che i vettori non debbano essere sempre ricalcolati. Se trovo più generalizzazioni minimali posso usare un qualche criterio di utilità per scegliere la migliore tabella da restituire ad esempio calcolando una loss tra la tabella orginale e quelle trovate ed andare a selezionare quella con loss minore.

## Algoritmo Incognito

Algoritmo di tipo bottom-up nato dopo quello della Samarati, che sfrutta in maniera più efficiente la visita del reticolo di generalizzazione. Come caratteristica principale ha quella di costruirsi lui i set dei quasi-identifier, mentre per l'algoritmo di Samarati andavano passati come parametro del metodo. L'idea di base è che la $k$-anonymity rispetto ad un sottoinsieme proprio dei $QI$ è una condizione necessaria, ma non sufficiente per la $k$-anonymity del set $QI$ e questa proprietà viene sfruttata in questo modo:

- **Iterazione** 1: verifica della $k$-anonimità per ogni singolo attributo che compone $QI$, scartando le generalizzazioni che non la soddisfano;
- **Iterazione** 2: combino a coppie tutte le possibili generalizzazioni che mi rimangono dall'iterazione precedente e verifico che soddisfino la $k$-anonymity, andando a scartare tutte le coppie di generalizzazioni che non la soddisfano;
- **Iterazione** $i$: combino tutte le tuple di $i-1$ attributi che soddisfino la $k$-anonymity in modo da avere tutti i set di $i$ attributi possibili su cui controllare nuovamente l'anonimità. Ad ogni passo quindi mi costruisco un reticolo con di generalizzazione ma con solo le combinazioni di generalizzazioni che al passo precedente si sono dimostrate $k$-anonime;
- **Iterazione** $|QI|$: restituisci i risultati ottenuti.

Questo approccio è lo stesso del metodo **a priori** del ML per l'estrazione degli itemset frequenti (Rule Model). Incognito sfrutta due reticoli, quello di aggregazione dei attributi e quello classico di generalizzazione che ottiene completamente solamente all'ultimo passo. Incognito usa le proprietà:

- **proprietà di generalizzazione**: se in uno dei reticoli un nodo soddisfa la $k$-anonymity allora la soddisferà qualsiasi suo nodo antenato composto da generalizzazioni dei suoi attributi. Questa proprietà mi permette di velocizzare molto ogni check delle iterazioni precedentemente descritte;
- **proprietà dei subsets**: quella usata da a priori, se la $k$-anonymity non è soddisfata per un certo set di attributi non potrà esserlo per ogni suo

superset perchè vado ad aggiungere più informazioni alla tabella.

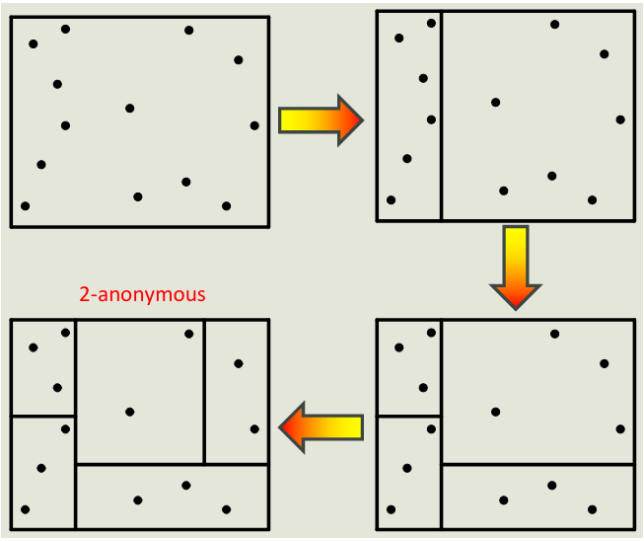
Nonostante le differenze con l'algoritmo di Samarati, avendo gli stessi obiettivi trova risulti molto simili.

## Algoritmi nello Spazio

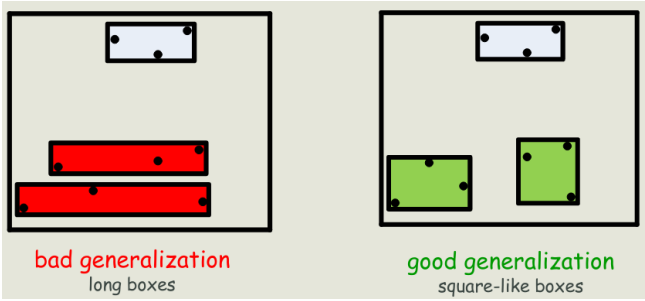
A differenza dell'algoritmo di Samarati ed Incognito che cercano nel reticolo di generalizzazione, vedremo ora degli algoritmi che ricercano nello spazio multi-dimensionale: ogni attributo che compone il $QI$ è una dimensione. In questo modo ogni tupla della tabella privata ($PT$) è un punto in questo spazio multi-dimensionale. Avrò associata anche il numero di occorrenze di tuple con gli stessi valori dei quasi-identifiers. Definito questo vediamo come si comportano i due algoritmi.

### Algoritmo di Mondrian

Lo spazio multidimensionale viene partizionato facendo degli split sulle dimensioni in modo tale che ogni regione dello spazio contenga almeno $k$ punti. Ogni punto della regione verrà rappresentato da un solo vettore che è una generalizzazione minimale dei punti della regione. Con questo algoritmo non vado quindi prima a generalizzare e poi controllare che la tabella sia $k$ anonima, ma partiziono lo spazio in modo tale che questa condizione sia soddisfatta. Il principio usato nel partizionamento è la **discernability metric** che penalizza i gruppi in base alla loro grandezza, questo perchè il partizionamento viene fatto con l'idea di avere in ogni area esattamente $k$ tuple, non di più che perdo di utilità. Quello che si fa è partizionare iterativamente i vari attributi e lo spazio $n$ dimensionale è come se fosse diviso in iper-rettangoli. Rispetto all'algoritmo di Samarati posso ottenere un partizionamento più fine che non intacchi sempre tutta la dimensione.

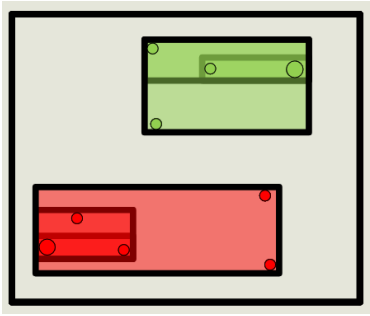


Questo algoritmo è molto buono nel trovare partizionamenti con esattamente il valore $k$ di record richiesti, ma posso ottenere gruppi con meno interpretabilità poichè caratterizzati da un iper-spazio molto grande.

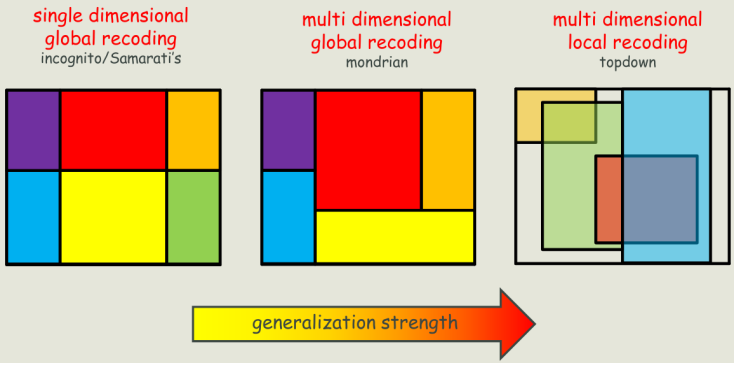



### Algoritmo Top-Down

L'algoritmo top-down cerca di risolvere il problema della creazione di iper-rettangoli lunghi e stretti andando ad usare la **normalized certainty penalty metric** che non prende in considerazione solo la cardinalità dei gruppi che crea. Questa metrica a parità di area predilige zone con perimetro più basso, preferendo quindi iper-cubi ad iper-rettangoli. In particolare possiamo dire che mentre Incoglito e Samarati fanno un partizionamento a livello globale dato che tagliano ogni dimensione completamente su tutto il dataset, Top-Down e Mondrian risultano approcci locali. Nello specifico questo algoritmo è ancora più locale di Mondrian nel senso che non va a partizionare lo spazio, ma cerca gruppi di tuple vicine. Questo algoritmo divide in due il dataset originale e continua a farlo fino a quando i gruppi non contengono un numero di tuple inferiore a $2k - 1$. L'algoritmo di split va a trovare prima **i semi** ossia i 2 punti nel dataset che sono i più distanti nello spazio e non ancora assegnati ad un gruppo. Questi due punti possono essere trovati in maniera esaustiva o approssimata con euristica. A questo punto i 2 semi diventeranno i due gruppi di split. Vengono quindi esaminati in maniera randomica i punti ad essi locali e vengono aggiunti al gruppo rispettando la minor espansione possibile del perimetro.



Questo algoritmo può portare a generalizzazione anche con sovrapposizioni ossia gruppi che condividono tra di loro valori degli attributi. Questo algoritmo tra tutti quelli visti è che quello che porta a gruppi più fini possibili e quindi migliori generalizzazione. Il problema è che mi può portare a punti di difficile interpretabilità: punti che ricadono in più aree possono essere rappresentati con generalizzazioni diverse sui quasi-identifiers.




## Limiti della $k$-Anonymity

Nonostante l'obiettivo della $k$-anonymity sia quello di scollegare l'attributo sensibile dai quasi-identifiers posso avere gruppo di tuple di dimensione $k$ (che rispetta la definizione di $k$-anonimità) che coincidono non solo sui valori dei quasi-identificatori, ma anche sull'attributo sensibile. Se quindi io conoscessi i valori dei quasi-identificatori di una persona perchè pubblici e so che il suo record è nella tabella noterei nella tabella anonima rilasciata che appartiene ad un gruppo di individui con tutti la stessa caratteristica, non ho risolto nulla praticamente. Quello appena descritto è un possibile attacco che può essere condotto che prende il nome di **homogenity attack**.

| id | Zipcode | Sex | National. | Disease |
|---|---|---|---|---|
| ... | ... | ... | ... | ... |
| 9 | 13053 | 31 | American | Cancer |
| 10 | 13053 | 37 | Indian | Cancer |
| 11 | 13068 | 36 | Japanese | Cancer |
| 12 | 13068 | 35 | American | Cancer |

| id | Zipcode | Sex | National. | Disease |
|---|---|---|---|---|
| ... | ... | ... | ... | ... |
| 9 | 130** | 3* | * | Cancer |
| 10 | 130** | 3* | * | Cancer |
| 11 | 130** | 3* | * | Cancer |
| 12 | 130** | 3* | * | Cancer |

E' possibile inoltre fare, in alcuni casi, un **background knowledge attack** basato sul fatto che l'attancate riesce ad inferire il corretto valore dell'attributo sensibile di un individuo grazie ad una conoscenza a priori che potrebbe avere. E' in sostanza la generalizzazione dell'attacco precedente e permette:

- **disclosure positiva**: pubblicando la tabella generalizzata $GT$ derivata

da quella privata $PT$ l'attaccante riesce ad identificare il valore corretto di un attributo sensibile con alta probabilità;

- **disclosure negativa**: l'attaccante riesce ad eliminare con alta probabilità valori ammissibili dell'attributo sensibile di un individuo.

Sono ovviamente entrambi da evitare dato che l'obiettivo malevole potrebbe proprio quello di capire l'assenza di un determinato attributo, come in questo caso la presenza o meno di un cancro:

| Id | Zipcode | Age | National. | Disease |
|---|---|---|---|---|
| 1 | 13053 | 28 | Russian | Heart Disease |
| 2 | 13068 | 29 | American | Heart Disease |
| 3 | 13068 | 21 | Japanese | Viral Infection |
| 4 | 13053 | 23 | American | Viral Infection |
| ... | ... | ... | ... | ... |

| Id | Zipcode | Age | National. | Disease |
|---|---|---|---|---|
| 1 | 130** | <30 | * | Heart Disease |
| 2 | 130** | <30 | * | Heart Disease |
| 3 | 130** | <30 | * | Viral Infection |
| 4 | 130** | <30 | * | Viral Infection |
| ... | ... | ... | ... | ... |

Per concludere tutti questi problemi sono causati da una "*cattiva*" distribuzione dei valori sensibili nei data subject.

# Beyond k-Anonymity

## Lack of Diversity

### Definizioni Utili

Diamo delle definizioni di base utili per presentare i prossimi concetti:

- $PT$ è la tabella privata di partenza $GT$ è la tabella generalizzata ottenuta da $PT$ e pubblicata;
- se $q$ è il valore di un attributo $Q$ non sensibile (e quindi quasi-identifier) in $PT$ allora il corrispondente valore presente in $GT$ sarà $q^\star$;
- se $s$ è un valore sensibile dell'attributo $S$ allora viene indicato con $n(q^\star, s)$ il numero di tuple $t^\star \in GT$ identiche per un certo valore sensibile ossia con $t^\star[Q] = q^\star$ e $t^\star[S] = s^\star$;
- viene detto $q^\star$**-block** o **classe di equivalenza** l'insieme delle tuple che sono state generalizzate in $GT$ assegnando il valore $q^\star$ al valore del quasi-identifier $Q$ di $PT$.

La **lack of diversity** è il nome formale data alla motivazione che rende possibile un *homogenity attack* e un *background knowledge attack*. Nello specifico l'assenza di diversità avviene per un attributo sensibile $S$ quando $\forall s' \neq s \quad t.c. \;\; n(q^\star, s') << n(q^\star, s)$ ossia quando preso un valore per un quasi-identifier il numero delle tuple che hanno $s$ come valore sensibile è molto più grande di qualsiasi altro valore possibile. In questo caso infatti abbiamo che $P(t^\star[S] = s \mid t^\star[Q] = q^\star) \approx 1$ e quindi con alta probabilità riesco ad inferire il valore dell'attributo sensibile semplicemente sapendo a quale classe di equivalenza l'individuo appartenga.

## $l$-diversity

Il problema della *lack of diversity* può essere superato con il **principio della** $l$**-diversity** che deve essere tenuto in considerazione nel processo di anonimizzazione di una tabella e che ci permette di rafforzare ulteriormente la privacy delle informazioni in maniera indipendente dall'algoritmo di $k$-anonimizzazione scelto. Per assicurare la **diversità** di un $q^\star$-block questo deve avere almeno

$l \geq 2$ valori differenti dell'attributo sensibile con all'incirca la stessa frequenza all'interno della classe di equivalenza. Ovviamente nel blocco ci possono essere più di $l$ valori, l'importante però è che gli $l$ più frequenti abbiano frequenza simile. Il parametro $l$ viene stabilito dal curatore del dataset che si preoccupa di anonimizzarlo. Se questo si verifica possiamo dire che il $q^\star$-block è **ben rappresentato** da $l$ **valori sensibili**. Questo principio non rende impraticabile gli attacchi precedenti, ma serve per ridurre drasticamente la loro probabilità di riuscita. Se avviene un *backgorund knowledge attack* per portare a termine con successo una *positive disclosure* l'attaccante ha bisogno di $l-1$ ulteriori informazioni sensibili del soggetto, in modo da riuscire ad identificarlo nel blocco. Generalizzando il principio possiamo dire che una tabella è $l$-diversa se ogni $q^\star$-block è $l$-diverso.

Garantire la $l$-diversity nel processo di $k$-anonimizzazione significa scegliere alla fine del processo di $k$-anonimizzazione la tabella nel reticolo, tra tutte le soluzioni possibili, quella che è anche $l$-diversa. Per guidare invece la ricerca verso tabelle $l$-diverse e non fare la selezione solo alla fine dobbiamo utilizzare la misura di **entropy $l$-diversity** come criterio per potare parti del reticolo di generalizzazione. Possiamo dire che una tabella generalizzata $GT$ è **entropy $l$-diversity** se per ogni $q^\star$-block:

$$-\sum_{s\in S} P(q^\star, s)\cdot \log(P(q^\star, s)) \geq \log(l) \qquad \text{con} \quad P(q^\star, s) = \frac{n(q^\star, s)}{\sum\limits_{s\in S} n(q^\star, s)}$$

Viene introdotta questa misura poichè gode della proprietà di **monotonicità**, cioè se una tabella generalizzata $GT$ soddisfa la entropy $l$-diversity allora la soddisfa anche ogni generalizzazione di $GT$. Questa proprietà unita con la **monotonicità** della k-Anonymity risulta fondamentale nell'algoritmo di ricerca, poichè esattamente come scarto un nodo e tutti i suoi discendenti nel reticolo se non soddisfano la $k$-anonymity posso nello stesso istante controllare e scartare per la $l$-diversità, questo soprattutto per algoritmi classici di ricerca nel reticolo come Samarati ed Incognito. Negli algoritmi di Mondrian e TopDown deve essere verificata in un altro modo.

4-anonymous data

| Id | Zipcode | Age | National. | Disease |
|---|---|---|---|---|
| 1 | 130** | <30 | * | Heart Disease |
| 2 | 130** | <30 | * | Heart Disease |
| 3 | 130** | <30 | * | Viral Infection |
| 4 | 130** | <30 | * | Viral Infection |
| 5 | 1485* | ≥40 | * | Cancer |
| 6 | 1485* | ≥40 | * | Heart Disease |
| 7 | 1485* | ≥40 | * | Viral Infection |
| 8 | 1485* | ≥40 | * | Viral Infection |
| 9 | 130** | 3* | * | Cancer |
| 10 | 130** | 3* | * | Cancer |
| 11 | 130** | 3* | * | Cancer |
| 12 | 130** | 3* | * | Cancer |

4-anonymous / 3-diverse data

| Id | Zipcode | Age | National. | Disease |
|---|---|---|---|---|
| 1 | 1305* | ≤40 | * | Heart Disease |
| 4 | 1305* | ≤40 | * | Viral Infection |
| 9 | 1305* | ≤40 | * | Cancer |
| 10 | 1305* | ≤40 | * | Cancer |
| 5 | 1485* | >40 | * | Cancer |
| 6 | 1485* | >40 | * | Heart Disease |
| 7 | 1485* | >40 | * | Viral Infection |
| 8 | 1485* | >40 | * | Viral Infection |
| 2 | 1306* | ≤40 | * | Heart Disease |
| 3 | 1306* | ≤40 | * | Viral Infection |
| 11 | 1306* | ≤40 | * | Cancer |
| 12 | 1306* | ≤40 | * | Cancer |

### Problemi $l$-diversity

Abbiamo essenzialmente 3 diversi problemi:

- All'aumentare del valore di $l$ comincio ad avere nel blocco moltissimi valori diversi dell'attributo sensibile ed inizio a perdere la possibile correlazione tra i valori dei quasi-identifier e l'attributo sensibile che è solitamente lo scopo dell'analisi statistica. Come caso limite avrò in ogni classe di equivalenza tutti i valori possibili dell'attributo sensibile, uno per tupla, perdendo completamente di significatività il dataset. E' per questo che il corretto valore del parametro $l$ va scelto accuratamente;
- Per definizione la $l$-diversity cerca di rendere tutti gli $l$ diversi valori dell'attributo sensibile ugualmente frequenti all'interno di ogni classe di equivalenza e ciò potrebbe portare a dataset con anomalie statistiche, causate dalla modifica delle probabilità reali di occorrenza di fenomeni rari o certi (come HIV). L'attacco in questione è chiamato **Skewness Attack** che può accadere quando la distribuzione del valore sensibile in un blocco si allontana molto da quella effettiva nella popolazione;
- Possiamo avere una tabella $k$-anonima e $l$-diversa, ma con un $q^\star$-block in cui i valori dell'attributo sensibile sono ovviamente sintatticamente diversi, ma semanticamente simili. In questo caso indipendentemente dal valore effettivo un attaccante riesce comunque ad ottenere informazioni sensibili su un individuo, sapendo che ricade in un certo blocco. L'attacco in questione è chiamato **Similarity Attack** ed è un problema generale della $k$-anonymity non superato con la $l$-diversity.

| id | Zipcode | Age | National. | Disease |
|---|---|---|---|---|
| 1 | 130** | <30 | * | Stomach ulcer |
| 2 | 130** | <30 | * | Stomach ulcer |
| 3 | 130** | <30 | * | Gastritis |
| 4 | 130** | <30 | * | Gastritis |
| ... | ... | ... | ... | ... |

## $t$-closeness

Una classe di equivalenza gode della $t$-closeness se la distanza tra la distribuzione di un attributo sensibile nella classe e la distribuzione di tale attributo nella tabella originaria non supera una soglia $t$. Una tabella viene quindi detta **$t$-close** se lo sono tutte le sue classi di equivalenza. Per capire questo concetto dobbiamo stabilire come calcolare la distanza tra due diverse distribuzioni **P** e **Q** calcolate su dati e quindi persone diverse. Richiedere che due distribuzioni di questo tipo abbiano una piccola distanza, e quindi $t$ piccolo, vuol dire andare ad annullare l'informazione rilevante che ci permette di trovare correlazione tra quasi-identifier

e attributi sensibili. Stiamo chiedendo infatti che ogni blocco abbia all'incirca la stessa distribuzione di valori e quindi la tabella generalizzata potrà risultare inutilizzabile a fini statistici se $t$ è molto piccolo. Come pro abbiamo però che riduciamo praticamente a zero la possibilità di uno Skewness Attack. La scelta del corretto valore di $t$ che sia un buon trade-off tra utilità e privacy è molto difficile. Date due distribuzioni $\mathbf{P} = \{p_1, p_2, ..., p_m\}$ e $\mathbf{Q} = \{q_1, q_2, ..., q_m\}$ possiamo usare la:

- **Variational distance** la più semplice tra le distanze definita come:

$$D_v[\mathbf{P}, \mathbf{Q}] = \sum_{i=1}^{m} \frac{|p_i - q_i|}{2}$$

- **Kullback-Leibler distance** definita come:

$$D_{KL}[\mathbf{P}, \mathbf{Q}] = \sum_{i=1}^{m} p_i \cdot \log(\frac{p_i}{q_i})$$

- **Earth Mover distance** usata dalla $t$-closeness in quanto le precedenti non tengono in considerazione la semantica degli attributi e non permettono di considerare più o meno significative le differenze di masse di distribuzione tra diversi valori. Questa distanza si basa sul concetto di minima quantità di lavoro necessaria per trasformare una delle due distribuzioni nell'altra, muovendo masse di distribuzione tra di loro. Abbiamo due diverse implementazioni di questa distanza a seconda che:
  - **attributo sensibile numerico**: con valori ordinabili $\{v_1, v_2, ..., v_m\}$ la distanza normalizzata tra due possibili valori numerici è $dist(v_i, v_j) = \frac{|i-j|}{m-1}$ e chiamando $r_i = \frac{p_i - q_i}{m-1}$ abbiamo:

$$D_{EM}[\mathbf{P}, \mathbf{Q}] = \sum_{i=1}^{m} |\sum_{j=1}^{i} r_j| = \frac{1}{m-1} \cdot (|r_1| + |r_1 + r_2| + \cdots + |r_1 + r_2 + ...r_{m-1}|)$$

  - **attributo sensibile categorico**: se i valori sono nominali o categorici possiamo ancora distinguere:
    - **dominio flat**: dove non c'è differenza tra un valore ed un altro dell'attributo sensibile (come colori occhi). In questo caso si usa la variational distance;
    - **dominio gerarchico**: dove esiste una tassonomia di valori come lo ZIP code, allora abbiamo:

$$D_{EM}[\mathbf{P}, \mathbf{Q}] = \sum_{N} cost(N)$$

      dove $N$ è un nodo interno e $cost(N)$ è il costo di muovere i rami dei suoi figli tra di loro.

La $t$-clossness è principio molto usato per evitare *swekness* e *similarity attack*, ma è molto meno interpretabile della $k$-anonymity o della $l$-diversity come definizione.

microdata

| id | Zipcode | Age | Salary | Disease |
|---|---|---|---|---|
| 1 | 47677 | 29 | 3K | Gastric ulcer |
| 2 | 47602 | 22 | 4K | Gastritis |
| 3 | 47678 | 27 | 5K | Stomach ulcer |
| 4 | 47905 | 43 | 6K | Gastritis |
| 5 | 47909 | 52 | 11K | Flu |
| 6 | 47906 | 47 | 7K | Bronchitis |
| 7 | 47605 | 30 | 8K | Bronchitis |
| 8 | 47673 | 36 | 9K | Pneumonia |
| 9 | 47607 | 32 | 10K | Stomach ulcer |

3-anonymous data

| id | Zipcode | Age | Salary | Disease |
|---|---|---|---|---|
| 1 | 4767* | ≤40 | 3K | Gastric ulcer |
| 2 | 4767* | ≤40 | 5K | Stomach ulcer |
| 3 | 4767* | ≤40 | 9K | Pneumonia |
| 4 | 4790* | >40 | 6K | Gastritis |
| 5 | 4790* | >40 | 11K | Flu |
| 6 | 4790* | >40 | 8K | Bronchitis |
| 7 | 4760* | ≤40 | 4K | Gastritis |
| 8 | 4760* | ≤40 | 7K | Bronchitis |
| 9 | 4760* | ≤40 | 10K | Stomach ulcer |

### Problema della $t$-closeness (e della $k$-anonymity)

Se ci sono più attributi sensibili da mascherare è possibile che un attaccante riesca ad identificare perfettamente una persona avendo a disposizione i suoi quasi-identifiers e uno dei valori sensibili. Questo è però un problema che caratterizza di per sè la $k$-anonymity che non prevede da parte di un attaccante la conoscenza di informazione sensibile. Il problema è che a livello pratico qualsiasi attributo può essere usato come quasi-identifier. Stiamo parlando di **membership disclosure** che è praticamente sempre applicabile se il dataset è $k$-anonimizzato dato che la soppressione o la generalizzazione di un quasi-identifier non ne altera la distribuzione nella popolazione originaria, non stiamo cambiando i valori reali del record.

## Membership Disclosure

Spostiamo la nostra attenzione dal non riuscir ad identificare il corretto record nel dataset per una persona e fare quindi attribute disclosure al non riuscir a capire se una persona vi appartenga o meno. Se ho db del reparto oncologico, sapere se una persona vi appartiene è già di per sè problematico.

### $\delta$-presence

Questa definizione rimane nel framework della $k$-anonymity e quindi della generalizzazione o soppressione di tuple. Data una tabella pubblica $T$ ed una privata $PT$, diciamo che la proprietà di **$\delta$-presence** vale per la generalizzazione $GT$ di $PT$ con $\delta = (\delta_{min}, \delta_{max})$ intervallo di valori se:

$$\delta_{min} \leq P(t \in PT \mid GT) \leq \delta_{max} \qquad \forall t \in T$$

In particolare in un dataset di questo tipo ogni tupla $t \in T$ è $\delta$-presente in $PT$, dove $\delta$ è un range che io definisco accettabile di probabilità di identificazione.

La presenza del bound superiore serve per evitare la positive disclosure, mentre quello del bound inferiore la negative disclosure. La cosa interessate è che la $\delta$-presence è una **proprietà monotonica**: se io ho due generalizzazioni $GT_1$ e $GT_2$ tale che $PT \leq GT_1 \leq GT_2$ allora se $GT_1$ è $(\delta_{min}, \delta_{max})$-present allora lo è anche $GT_2$. Può quindi essere usata per potare il reticolo delle generalizzazione durante la ricerca dato che se una tabella non è $\delta$-present allora non lo saranno nessuna delle sue specializzazioni.

#### Scelta di un buon $\delta$

La scelta di una buona coppia di probabilità avviene mediante l'uso di credenza a priori, di credenza a posteriori dopo la pubblicazione di $GT$ e di informazioni sulla tabella privata $PT$ e sulla generalizzata $GT$. Se abbiamo ad esempio una tabella privata $PT$ con pazienti oncologici e viene pubblicata una sua versione generalizzata $GT$, allora abbiamo ad esempio che: la credenza a priori è la probabilità che una persona abbia il cancro all'interno della popolazione originale, mentre la credenza a posteriori è la probabilità che una persona abbia il cancro una volta venuti a conoscenza dell'informazione contenuta in $GT$. Altri fattori che possono essere presi in considerazione potrebbero essere la differenza del numero di tuple tra $PT$ e $GT$ così come altre informazioni specifiche dello scenario.

## Conclusioni sulla $k$-anonymity

Nonostante sia fondamentale fare un trade-off tra utilità del dataset e anonimità, l'utilità non viene presa in considerazione in maniera esplicita nell'algoritmo, generalmente si spera che l'analisi dei dati ottenuti alla fine del processo sia ancora possibile, ma questo non sempre è garantito. Senza contare che trovare la generalizzazione $k$-minimale con soppressione è NP-hard e spesso vi è anche un trade-off tra efficienza ed efficacia dell'algoritmo di anonimizzazione. La $k$-anonymity è inoltre pensata originariamente per la pubblicazione di una singola tabella di microdata anche se ovviamente un db è formato da molte tabelle anche collegate tra loro con link di chiave esterna. In questo scenario il processo si complica parecchio cosi come quando non vi è più un solo record per individuo, ma ce ne sono di più. Un altro grosso problema che intacca la $k$-anonymity così come tutti gli algoritmi che lavorano sui dati è la **curse of dimensionality** ovvero il fatto che all'aumentare della dimensione e in questo caso degli attributi per ogni record, le distanze si appiattiscono così come le somiglianze e comincio ad avere vettori molto sparsi e diversi.

# Differential Privacy

## Motivazioni

Dato l'aumento della mole di dati che le aziende private e pubbliche continuano a raccogliere nasce l'esigenza sia di condividere i dati tra i vari attori in modo da ottenere analisi più precise ed accurate, sia di risolvere i problemi di privacy che la condivisione può portare a causa delle informazioni sensibili sugli individui contenute al loro interno. Per conciliare queste due spinte opposte nel 2006 Cynthia Dwork e altri ricercatori definirono la **differential privacy** come strumento che garantisce la non violazione dell'identità dei data subjects che affidano le proprie informazioni ad un curatore fidato del dataset, nè mediante la pubblicazione dei risultati delle analisi nè con il linking attack. La differential privacy permette anche di evitare l'identificazione dei records da parte di un attaccante che possiede già parte del dataset originale e che vorrebbe inferire l'informazione sulla parte a lui mancante. Come caso limite di questo scenario abbiamo che l'avversario ha tutto il dataset del curatore tranne un record e vuole capire a quale data subject appartenga. La differential privacy vuole assicurare che se anche il dataset del curatore contesse un solo record in più di quello dell'attaccante quest'ultimo non sarebbe in grado di capirlo. A livello teorico l'idea alla base della differential privacy è quella di non pubblicare il risultato esatto di una query, ma andare invece ad estrarre il valore da restituire da una distribuzione di probabilità centrata sul valore reale. In questo modo l'attaccante vedendo il valore della query ad esempio non riesce a capire da quale distribuzione sia stato campionato. Questo viene chiamato **meccanismo randomizzato**. Formalmente possiamo dire che dato l'insieme di tutti i datasets $\Omega$ con certo schema, un certo codominio $\mathcal{R}$ dei valori restituiti e $\mathcal{Q}$ l'insieme di tutte le queries da $\Omega$ a $\mathcal{R}$, allora se $q \in \mathcal{Q} : \Omega \to \mathcal{R}$ e $D \in \Omega$ il meccanismo randomizzato è una funzione stocastica $\mathcal{M} : \Omega \times \mathcal{Q} \to \mathcal{R}$ in cui se $q(D)$ è il risultato vero $\mathcal{M}$ restituisce $\widetilde{q(D)}$. Il cuore della differential privacy è proprio quello della definizione di un meccanismo randomizzato che mi permetta di usare queries stocastiche, che ritornano il valore reale più del rumore che segue una certa distribuzione.

## $\epsilon$-differential privacy

Dati $\Omega$ e $\mathcal{Q}$ con la definizione precedente e $D$ e $D'$ due **dataset adiacenti** ossia che sono identici a meno di un record $D' = D \cup \{d\}$ (o viceversa) la definizione formale di differential privacy pura è:

> Un meccanismo $\mathcal{M} : \Omega \times \mathcal{Q} \to \mathcal{R}$ rispetta la **$\epsilon$-differential privacy** con $\epsilon > 0$ **privacy budget** se $\forall q \in \mathcal{Q},\ \forall (D, D') \in \Omega \times \Omega$ adiacenti e $\forall r \in \mathcal{R}$
>
> $$e^{-\epsilon} \leq \frac{P(\mathcal{M}(D, q) = r)}{P(\mathcal{M}(D', q) = r)} \leq e^{\epsilon}$$

Il privacy budget indica il livello di privacy che viene rispettato dal meccanismo, infatti quando $\epsilon \to 0$ $e^{-\epsilon} \approx e^{\epsilon} \approx 1$ ossia stiamo chiedendo che il rapporto valga 1 e quindi che le probabilità siano identiche. Più è piccolo $\epsilon$ in valore assoluto, più forti sono le garanzie di privacy. A livello grafico di distribuzioni di probabilità $\epsilon$ è la distanza massima che deve separare le due curve per ogni possibile valore del codominio. Ovviamente noi vorremmo un mecccanismo privato *accurato* ossia che restituisca un valore vicino a quello reale. Questo però non è garantito dalla definizione appena introdotta, si può infatti costruire un meccanismo che rispetti la definizione, ma che sia completamente inutilzzzabile.

### Meccanismo di Laplace

Il meccanismo più famoso ed utilizzato è sicuramente il **meccanismo di Laplace** che dato un dataset $D$ e query $q$ costruisce una distribuzione di Laplace$(\mu, \sigma)$ centrata in $q(D)$ da cui estrarre il valore da restituire. La distribuzione di Laplace ovviamente non è unica, ma è una famiglia che dipende da due parametri: $\mu$ che indica dove è centrata, nel nostro caso $q(D)$; e $\sigma$ che esprime la forma esatta della distribuzione. Tanto più $\sigma$ è piccolo tanto più la distribuzione è appuntita sul valore centrale, ma tanto più distanti risulteranno le due distrubuzioni su dataset adiacenti; mentre tanto più è grande più diventa una sorta di uniforme appiattita priva di utilità, ma priva anche di differenze tra distribuzioni adiacenti. Al fine dell'utilità $\sigma$ deve essere quindi il più piccolo possibile, mentre per la privacy il più grande possibile. La scelta del corretto $\sigma$ dipenderà quindi sia da $\epsilon$ che da $q$, la query che vogliamo distorcere. Il parametro deve essere definito in modo che vada bene per ogni coppia di dataset adiacenti. Abbiamo quindi che $\epsilon$ e $\sigma$ sono inversamente proporzionali, mentre $\sigma$ è direttamente proporzionale alla distanza dei risultati ottenuti con la query $q$ a seconda della coppia dei dataset adiacenti: $\sigma = GS_1(q)/\epsilon$.
Otteniamo lo stesso risultato se invece che campionare il nuovo risultato da restituire campioniamo il rumore che deve essere sommato all'originale, usando $Lap(0, GS_1(q)/\epsilon)$:

$$\mathcal{M}_{Lap}(q, D) = q(D) + Lap(0, \frac{GS_1(q)}{\epsilon})$$

Funziona allo stesso modo anche quando il risultato di $q(D)$ non è scalare, ma è un vettore.

### Sensitività Globale

La sensitività globale $GS_1(q)$ di una query $q$ è la misura di massima variazione che posso ottenere come risultato di $q$ presi due dataset adiacenti $D$ e $D'$:

$$GS_1(q) = \max_{(D,D')} \| q(D) - q(D') \|_1$$

Si dimostra che il meccanismo di Laplace che imposta $\sigma = GS_1(q)/\epsilon$ rispetta la $\epsilon$-differential privacy ed ha i migliori valori dei parametri ai fini dell'utilità.

### Al di là del meccanismo di Laplace

Il meccanismo di Laplace non può essere utilizzato quando:

- **Queries non numeriche**: è privo di senso sommare numero a valore categorico;
- **Selezione differenzialmente privata**: data una query numerica $q$ che restituisce il valore che massimizza una certa funzione di utilità è possibile che se il valore restituito venga offuscato con rumore, non sia più quello corretto che massimizza l'utilità desiderata.

## Mecanismo Esponenziale

Per risolvere questi problemi possiamo usare il **meccanismo esponenziale**. Supponiamo che $q$ sia la query che deve restituire il valore $r \in \mathcal{R}$ in modo tale da massimizzare una funzione di utilità $U : \Omega \times \mathcal{R} \to \mathbb{R}$, allora il meccanismo esponenziale associa ad ogni valore $r_i$ una probabilità di estrazione proporzionale al corrispondente valore di utilità $u_i$ ad esso associato per un certo dataset $D_j$ a cui appartiene. In questo caso la distribuzione esponenziale che andiamo a definire dipenderà ovviamente da $\epsilon$ per garantire la privacy, ma anche da $U$:

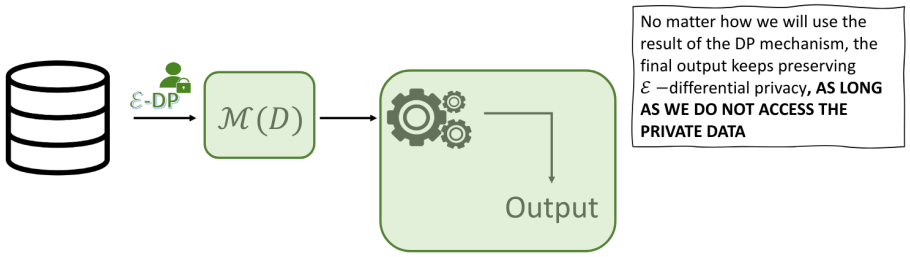
$$p(r_i) \propto e^{\frac{\epsilon \cdot U(D_j, r_i)}{2 \cdot GS_1(U)}} \qquad \text{con} \quad GS_1(U) = \max_{r \in R} \max_{(D,D')} |U(D, r) - U(D', r)|$$

## Proprietà della differential privacy

### Teorema di post-processing

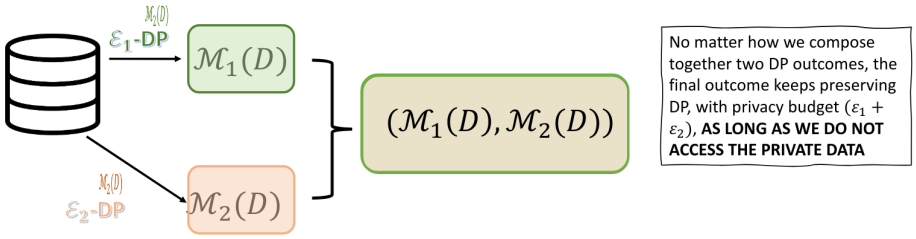
Dato un meccanismo $\mathcal{M} : \Omega \to \mathcal{R}$ che rispetta la $\epsilon$-differential privacy e una funzione $f$ che ha come dominio $\mathcal{R}$, possiamo dire che $f \circ \mathcal{M}$ rispetterà ancora la $\epsilon$-differential privacy. Ossia posso applicare al risultato di un meccanismo una qualsiasi operazione di post-processing che non abbia più necessità di fare

riferimento ai dati originali privati e avere la garanzia di rispettare la $\epsilon$-differential privacy. Conclusione: il post-processing non peggiora le garanzie di privacy.



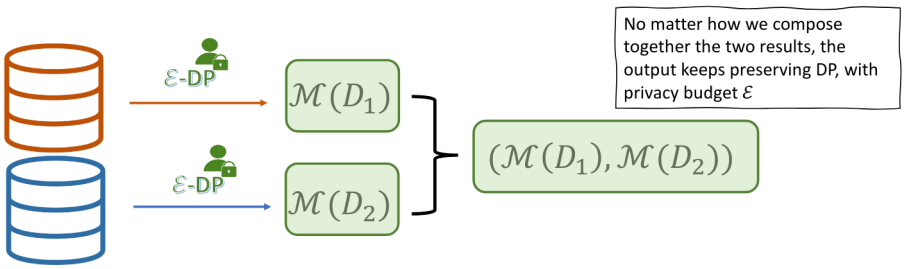

### Teorema di composizione

Dati due meccanismi diversi $\mathcal{M}_1$ e $\mathcal{M}_2$ che godono rispettivamente della $\epsilon_1$ e $\epsilon_2$ differential-privacy abbiamo che il meccanismo complessivo $\mathcal{M}$ che pubblica entrambi i risultati sul dataset $D$, ossia tale che $\mathcal{M}(D) = (\mathcal{M}_1(D), \mathcal{M}_2(D))$ rispetterà la $(\epsilon_1+\epsilon_2)$-differential privacy. Il meccanismo complessivo è ovviamente meno privato dei singoli dato che mostra più informazioni. Una volta ottenuti i risultati questi possono essere composti tra loro per il teorema precedente senza alcun problema.




### Teorema di composizione in parallelo

Dato un dataset $D$ originario lo andiamo a spezzare in due partizioni $D_1$ e $D_2$ per poi processare con lo stesso meccanismo $\mathcal{M}$ in maniera separata i due blocchi, abbiamo che il meccanismo complessivo delle due parti $\mathcal{N}(D) = (\mathcal{M}(D_1), \mathcal{M}(D_2))$ rispetta la $\epsilon$-differential privacy. Se usassi due meccanismi diversi con due parametri diversi, allora quello complessivo rispetterebbe il maggiore dei due $\epsilon_i$.




Questi tre teoremi ci sia di verificare il valore di $\epsilon$ rispettato da un meccanismo sia ci permettono di rendere privato un algoritmo molto complesso andandolo a spezzare in sottoparti.

## $(\epsilon, \delta)$-differential privacy

A livello pratico si è visto che la definizione originale di differential privacy pura può essere un po' troppo stringente e rischia di inserire troppo rumore ed avere risultati non molto utili. Si è quindi pensato di rilassare la definizione introducendo la $(\epsilon, \delta)$**-differential privacy**, dove $\delta \in (0, 1)$ è un **termine di tolleranza**. Un meccanismo rispetta la $(\epsilon, \delta)$-differential privacy se:

$$P(\mathcal{M}(q, D) = r) \leq e^{\epsilon} \cdot P(\mathcal{M}(q, D') = r) + \delta$$

Questo vuol dire che il meccanismo $\mathcal{M}$ soddisfa la $(\epsilon, \delta)$-differential privacy se esiste un evento $E$ con probabilità $P(E) \geq 1 - \delta$ tale che ogni volta che si verifica $E$ allora $\mathcal{M}$ rispetta la privacy. $\delta > 0$ è quindi una **probabilità di fallimento** del meccanismo che decidiamo di accettare/tollerare per garantire più utilità ai dati analizzati. Stiamo implicitamente trascurando il fatto di garantire che il rapporto tra le probabilità sia prossimo ad 1 per tutti i possibili valori $r$ anche per quelli molto improbabili. I teoremi visti prima continuano anche qua a valere con somma sia dei $\epsilon_i$ che $\delta_i$.
Per quanto riguarda la scelta del parametro $\delta$ questo deve essere sicuramente scelto in modo che sia piccolo, poichè in caso di fallimento potrebbero esserci conseguenze anche gravi. Inoltre il suo valore preciso dipende fortemente dal tipo di meccanismo che decidiamo di usare. E' facile mostrare che quando $\delta = 0$ ottengo la precedente definizione di $\epsilon$-differential privacy.

### Meccanismo di Gauss

Uguale al meccanismo di Laplace solo che la distribuzione di campionamento del rumore è Gaussiana:

$$\mathcal{M}_{Gauss}(q, D) = q(D) + \mathcal{N}(0, \sigma^2) \qquad \text{con} \qquad \sigma = \frac{2 \cdot GS_2(q)^2 \cdot \log(1.25/\delta)}{\epsilon^2}$$

La scelta indicata della deviazione standard ci garantisce il rispetto della $(\epsilon, \delta)$-differential privacy se $\epsilon \in (0, 1)$, mentre Laplace non aveva limitazioni nella scelta di $\epsilon$. Con sensibilità globale:

$$GS_2(q) = \max_{(D',D)} \parallel q(D) - q(D') \parallel_2$$

A livello di utilità il meccanismo di Laplace è migliore di quello di Gauss perchè ha picco più alto sul valor medio che risulta quindi più probabile e la probabilità scende anche più velocemente verso a 0 per valori lontati dal valor medio. Quindi a parità di $\epsilon$ e con $GS_1(q) = GS_2(q)$ il meccanismo di Laplace compromette meno i dati con rumore rispetto al meccanismo di Gauss.

## Scenario Non-Interattivo

Possiamo supporre di essere in due possibili scenari:

- **Interattivo**: il curatore fidato raccoglie i dati degli utenti e vuole mantenerli sicuri e va a pubblicare i risultati di 1 o + analisi sui dati che possiede in maniera differenzialmente privata. Questo è lo scenario fino ad ora ipotizzato e trattato. Viene chiamato interattivo poichè l'analista continuamente chiede statistiche/analisi al curatore sui dati che possiede il quale dovrà rispondere a modo. Il possibile problema di questo scenario è che se il curatore risponde a $n$ domande in maniera $\epsilon$-differenzialmente privata alla fine, per il teorema di composizione, avrà rispettato la $n\epsilon$-differential privacy. Il curatore deve quindi a priori scegliere il budget massimo da spendere che partizionerà su tutte le richieste;
- **Non Interattivo**: risolve il problema precedente in cui il curatore invece che aspettare e rispondere ad ogni domanda dell'analista, fa una copia del suo dataset in modo che rispetti la $\epsilon$-differential privacy e pubblica quella versione. Ogni analista per il teorema di post-processing potrà farne sopra tutte le analisi del caso e verrà sempre garantito il valore di privacy originario. Per fare ciò generalmente si può o creare un dataset sintetico il più simile possibile a quello originale ossia come se i record fossero estratti da stessa distribuzione sottostante oppure pubblicare una versione sintetica che rispetti le macro-statistiche complessive più importanti come media, varianza, ecc... Come ultima ipotesi vi è quella di rilasciare una **sinossi** del dataset privato ossia un insieme molto vasto e completo di analisi già fatte dal curatore e pubblicate per gli analisti.

## Curatore Non Fidato

Entrambi gli scenari precedenti prevedono la figura di un curatore fidato dalle persone a cui concedere le proprie informazioni private. Nel caso in cui questa figura non esistesse o equivalentemente non fosse fidata bisogna anticipare l'aggiunta

del rumore. I data subjects prima di inviare i loro dati reali li modificano con il valore stocastico. In questo scenario cambia però la definizione di differential privacy: mentre prima volevamo garantire che un attaccante non potesse capire la presenza o assenza di un individuo nel dataset, ora quell'informazione è nota, si sa quali e quante siano le persone che comunicano i dati. Vogliamo invece mascherare a tutti come è fatto un record di un individuo.

## $(\epsilon, \delta)$-local differential privacy

Si parla di differential privacy **locale** in quanto ogni utente distorce i propri valori. Sia $\mathcal{U}$ l'insieme di tutti i record allora possiamo dire che un meccanismo $\mathcal{M}$ preserva la $(\epsilon, \delta)$-local differential privacy se per ogni query $q : \mathcal{U} \to \mathcal{R}$, per ogni possibile coppia di record $x, x' \in \mathcal{U}$ e per ogni possibile risultato $r \in \mathcal{R}$:

$$P(\mathcal{M}(q, x) = r) \leq e^{\epsilon} \cdot P(\mathcal{M}(q, x') = r) + \delta$$

Si chiede quindi indistinguibilità, a livello probabilistico, di ogni possibile output $r$ di ogni queries se viene calcolato su un record $x$ o su ogni altro record $x'$. Non abbiamo più il concetto di **adiacenza**. Tutti i teoremi precedenti continuano a valere. Generalmente l'approccio locale aggiunge più rumore poichè stiamo rivelando più informazioni, mentre nell'approccio globale riveliamo solo il risultato dell'analisi sull'intero dataset qua potenzialmente possiamo rilasciare l'intero dataset distorto.

### Randomized Response

E' tra i meccanismi più noti per rendere una funzione localmente differenzialmente privata. E' una tecnica sviluppata già negli anni '70 per rilasciare dati ed in particolare per condurre studi ed analisi su informazioni private che non si volevano comunicare. L'idea è introdurre "*non sempre*" del rumore nel record di una persona che mente probabilisticamente. Un possibile attaccante che legge il record non sa se l'informazione è reale o se è così perchè la persona è stata costretta a mentire. Ad esempio per un'informazione binaria l'utente lancia una moneta:

- **croce**: dice il vero al 50%;
- **testa**: dice una risposta che è completamente slegata dal vero al 50%. Lanciando nuovamente la moneta restituirà uno dei due valori a seconda dell'esito.

Quando è nata la differential privacy si è scoperto che questo meccanismo la rispetta con un $\epsilon = \log\ 3$. Questo meccanismo ha come difetto il fatto che sia applicabile solo quando la query è categorica ed in particolare ha solo due valori possibili.

**Generalized Randomized Response**

Questo meccanismo generalizza il precedente e, data un query $q : \mathcal{U} \to \mathcal{R}$ con $|\mathcal{R}| = k$ possibili valori dell'attributo categorico, viene restituito come output il valore estratto dalla seguente distribuzione di probabilità è :

$$P(\mathcal{M}(x,q) = v) = \begin{cases} \frac{e^\epsilon}{e^\epsilon + k - 1} & \text{se } v \text{ è il vaore reale} \\ \frac{1}{e^\epsilon + k - 1} & \text{altrimenti} \end{cases}$$

Questo meccanismo rispetta la $\epsilon$-local differential privacy. Se $\epsilon \to 0$ allora le due probabilità coincidono e sarò in caso di distribuzione uniforme e restituirò spesso valori errati. Ritroviamo il classico comportamento di $\epsilon$: più è piccolo meno sono utili i dati, ma più sono protetti. Se $k = 2$ e $\epsilon = \log\ 3$ ci riconduciamo alla risposta randomizzata precedente. Il problema di questo meccanismo è che peggiora molto all'aumentare del valore di $k$, poichè perde accuratezza e può addirittura capitare che il valore reale abbia meno probabilità della somma delle probabilità associata agli altri valori. In questo caso i dati che sarebbero restituiti possono non essere sensati.

Per valori numerici e non categorici possiamo applicare i precedenti metodi di Laplace e Gauss.

# Differential Privacy nel Machine Learning

Negli ultimi anni sono stati pubblicati moltissimi algoritmi di machine learning che rispettano la definizione di differential privacy. *Che cosa vuol dire fare ML privato?*
Vuol dire sostanzialmente **allenare il modello in maniera differenzialmente privata** ossia costruire il corretto modello di ML con aggiunta di rumore in modo che dal modello finale non si possa capire se un certo record era stato usato o meno per il suo allenamento. Non ci interessa che il modello finale rispetti la local differential privacy e che quindi quando eseguito su un record sia impossibile ricondurlo a lui, ma siamo interessati all'allenamento che prende e maneggia direttamente i dati sensibili del training set. Già di base per il machine learning è fondamentale che un modello non dipenda troppo dal training set, al fine di evitare l'overfitting, quindi in questo senso il ML e la DP hanno obiettivi comuni. Nello specifico l'allenamento di un modello può essere visto come una funzione $A : \Omega \to \mathbb{R}^d$ che prende come input un dataset e restituisce un vettori di pesi per il modello. Dato questo possiamo avere 3 diversi tipi di meccanismi di differential privacy:

- **Input Perturbation**: consiste nell'aggiungere il rumore nel training set di partenza con meccanismi di differential privacy per poi fare l'allenamento standard su questo garantendo la $\epsilon$-differential privacy per il teorema di post-processing, questa è la tecnica più gettonata a livello pratico;
- **Output Perturbation**: introdurre il rumore solo alla fine, nel momento in cui ho i pesi perfetti li vado a perturbare. A primo impatto risulta la più

semplice tra le 3 procedure perchè basta applicare il meccanismo di Laplace al vettore dei pesi, ma per farlo devo conoscere la sensitività globale della funzione $A$ che è praticamente impossibile da calcolare con precisione: capire quale sia la variazione dei pesi a fronte di picccole perturbazioni del training set. L'altro modo che si potrebbe ipotizzare è: dato che l'algoritmo di discesa ottimizza una certa funzione di loss $\mathcal{L}$ si può pensare di usare un meccanismo esponenziale che assegni ad ogni vettore di pesi un'utilità in termini di loss e campioni probabilisticamente. Il problema è che i possibili valori di $\theta$ vettore dei pesi sono infiniti, senza contare la difficoltà di calcolare la sensitività globale della funzione di loss;
- **In-Process Mechanism**: introdurre il rumore nei passi dell'algoritmo di discesa del gradiente che vuole apprendere i migliori set di pesi. Dato che questi algoritmi sono iterativi possiamo vedere ogni iterazione come una sotto-funzione di quella complessiva $A$ di apprendimento di cui può essere più facile calcolarne la sensitività globale. Il teorema di composizione ci permetterebbe inoltre di sapere il livello di privacy rispettato alla fine del processo.

Vediamo un esempio.

# $k$-Means

## In-Process Perturbation

Vogliamo calcolare i $k$ centroidi in maniera differenzialmente privata, ricostruiamo i concetti base di questo algoritmo di clustering:

- **training set**: insieme di vettori $n$-dimensionali, ogni punto è un record/individuo;
- **modello**: funzione che preso in input un punto lo assegna al cluster il cui centroide ha con lui distanza euclidea minima;
- **algoritmo**: lo scopo dell'algoritmo è apprendere i migliori centroidi in modo da minimizzare la distanza euclidea di ogni punto del training set dal centroide del cluster a cui viene assegnato. Rispetto all'implementazione classicia del $k$-means andremo a restituire solo i centroidi e non la clusterizzazione del training set perchè ovviamente è un'informazione privata.

La procedura consiste nell'aggiungere del rumore di Laplace nel momento in cui calcolo, ad ogni iterazione, i centroidi dei clusters che ho costruito: non calcolo la vera media, ma la media distorta. Per il teorema di composizione in parallelo se calcolo ogni centroide in modo che sia $\epsilon$-differenzialmente privato alla fine dell'iterazione rispetterò la $\epsilon$-differential privacy dato che ogni centroide per il suo calcolo usa porzioni disgiunte del training set. La sensitività globale per il calcolo di un centroide dipende dallo spazio in cui possono variare i vettori: potenzialmente è $\infty$, ma se ogni dimensione assume solo valori in $[-1, 1]$ la sensitività globale della funzione che restituisce il numero di esempi in un cluster e la somma dei valori in ogni dimensione è $d + 1$. A questo punto per post-

processing posso ottenere il centroide semplicemente dividendo le somme per il numero di punti presenti nel cluster. Dato però il teorema di composizione in sequenza se il mio privacy budget complessivo che voglio rispettare è $\epsilon$ dovrò a priori stabilire il numero massimo di iterazioni $n$ in modo che ognuna abbia un budget di $\epsilon/n$. A livello pratico si è visto che questo meccanismo non risulta molto accurato, i centroidi che ottengo alla fine del processo risultano infatti abbastanza imprecisi, all'aumentare del numero di iterazioni si hanno distorsioni enormi.

### Input Perturbation

Questa procedura prevede il rilascio privato dei dati su cui posso poi applicare (per post-processing) una qualsiasi operazione io voglia. Funziona nel seguente modo:

- dato lo spazio di partenza in cui risiedono i punti, vado a dividerlo in una griglia di $m$ celle;
- ogni punto che ricade in una cella viene approssimato con il centro della cella stessa. Ovviamente più è grande $m$ più mantengo informazioni, ma ho meno privacy.

In questo processo le coordinate dei centri delle celle non devono essere sporcati nella pubblicazione poichè non dipendono dalle informazione private, cioè che invece va calcolato in modo differenzialmente privato è il conteggio dei punti che cadono in ogni cella e che quindi vengono approssimati con lo stesso centro. Questo si può fare facilmente con il meccanismo di Laplace con sensitività globale 1, dato che si tratta di un conteggio. Questo tecnica funziona meglio della precedente.

## Limiti e Aspetti Critici

Tra le problematiche che possiamo affrontare nella costruzione di un algoritmo differenzialmente privato come visto in Machine Learning and Differential Privacy ci sono:

- **Scelta del corretto** $\epsilon$: la possibilità di poter scegliere $\epsilon$ è una buona cosa perchè ci dà: flessibilità nell'applicazione di uno stesso meccanismo in contesti diversi, ci permette di dare una stima accurata del livello di privacy che stiamo rispettando e ci permette di confrontare diversi meccanismi proprio in base al valore di $\epsilon$ che usano. La scelta del corretto valore di $\epsilon$ è però molto difficile dato che dobbiamo fare un giusto trade-off tra riservatezza garantita da valori piccoli con utilità garantita invece da valori grandi. La stessa inventrice della DP afferma la difficoltà di questa scelta dato che dipende molto dai dati, dallo scopo dell'analisi, dal meccanismo usato, dall'algoritmo originale, ecc...;
- **Calcolo della** $GS$: il problema della sensitività globale è che dipende ovviamente dalla funzione che vogliamo distorcere sulla quale non abbiamo

alcun tipo di controllo e spesso può capitare che la $GS$ sia molto grande se non addirittura $\infty$ nel caso in cui dovessi scegliere un qualsiasi valore continuo. Se però la $GS$ è molto grande sarà elevata la quantità di rumore da aggiungere portando a dati che possono essere privi di utilità. A volte quindi si preferisce usare la sensitività locale $LS$, dato che la $GS$ viene calcolato come massima variazione tra tutte le possibili coppie di dataset adiacenti con un certo schema. Ma se noi siamo interessati a confondere il nostro dataset con uno a lui vicino possiamo calcolare la sensitività solo nel suo intorno di dataset adiacenti e non tra tutti i possibili costruibili:

$$LS_1(q, D) = \max_{D'} \ \| \ q(D) - q(D') \ \|_1$$

Nella definizione vado a fissare il dataset $D$ di cui calcolo il vicinato e ovviamente abbiamo che $LS_1(q, D) \leq GS_1(q)$ per ogni dataset di interesse. Il problema è però che se in un meccanismo sostituiamo la $GS$ con la $LS$ non rispettiamo più la definizione di DP, non tanto per il meccanismo in se, ma per il fatto che per ottenere la distribuzione Laplaciana dobbiamo rivelare l'informazione riguardo alla sensibilità locale del nostro dataset privato. Questo si può risolvere non andando a fissare i parametri della distribuzione, ma permettendo di cambiarli e calibrarli ogni volta a seconda del dataset che mi ritrovo, ma anche questo caso non va bene perchè ottengo curve di distribuzione troppo diverse l'una dall'altra a seconda del dataset locale che ho a disposizione. Un meccanismo ibrido è il **propose-test-release** la cui idea è quella di non dichiarare la $LS$, ma di fissarla come upper-bound in modo che sia $LS \leq U \leq GS$ e non dipenda da nessun dataset. Prende questo nome perchè è l'analista che propone il bound $U$ di utilità dei dati e il curatore per accettarlo dovrà verificare che sia compreso tra sensitività globale e quella locale. Bisogna ancora fare attenzione al calcolo del bound in modo che non si possa inferire la sensibilità locale di nessun dataset da esso;

- **Controllo del numero di iterazioni**: più sono le iterazioni dell'algoritmo meno privacy vado a rispettare, a meno di equo partizionamento a priori, il che prevede il controllo a priori del numero masssimo di iterazioni da compiere;
- **Problemi di Convergenza**: se algoritmi garantiscono convergenza nella loro forma classica non è detto che la garantiscano quando si decide di distorcerli;
- **Risultati fuori dal dominio**: se ho una semplice query numerica $q$ che resituisce valori in un intervallo ad esempio $(0, 1)$, l'aggiunta del rumore anche piccolo per valori estremi potrebbe portarci a valori non ammissibili. Generalmente in questo caso o si ripete il meccanismo fino a quando non si ottiene un valore accettabile, oppure si può proiettare tale valore nel dominio riducendolo al massimo se supera per eccesso e al minimo se supera per difetto che posso fare dato che è post-processing;
- **Scelta dei parametri/iperparametri**: la scelta del miglior valore dei parametri/iperparametri per un algoritmo è spesso difficile. In genere o si

usano conoscenze sui dati oppure si può fare fine-tuning, il problema è che in un setting privato spesso non ci è consentito di farlo.

# Privacy nei Sistemi Distribuiti

## Contesto

Siamo nel contesto di analisi di dati complessiva (sia statistica sia costruzione di modello di machine learning o altro ancora) in un sistema in cui abbiamo nodi che raccolgono dati diversi e vorrebbero che nessun altro nodo possa accedere alle proprie informazioni.

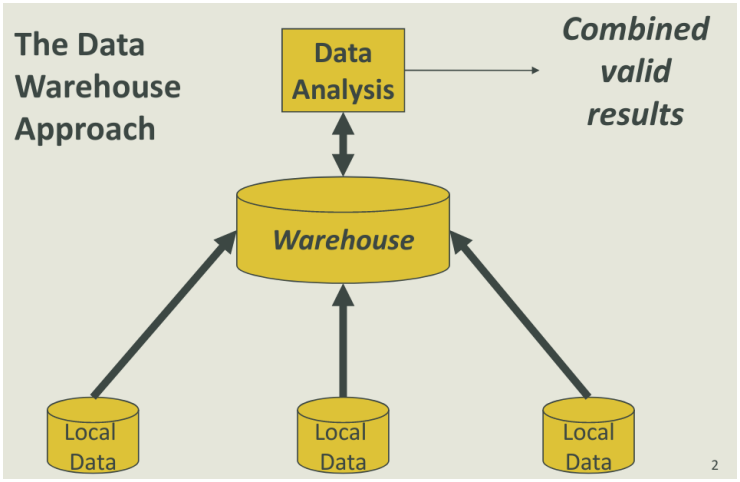
## Metodo Standard

Fino a poco tempo fa si usava l'approccio di **data warehouse**: i nodi che possedevano db locali con le informazioni da loro raccolte andavano a popolare un data warehouse che raccoglie i dati da tutti i nodi per aggregare e fare analisi. Il modulo di analisi si interfaccia poi direttamente con il data warehouse per chiedere ed ottenere informazioni sui dati. I dati si trovano quindi alla fine tutti all'interno del data warehouse centralizzato portando a diversi problemi:

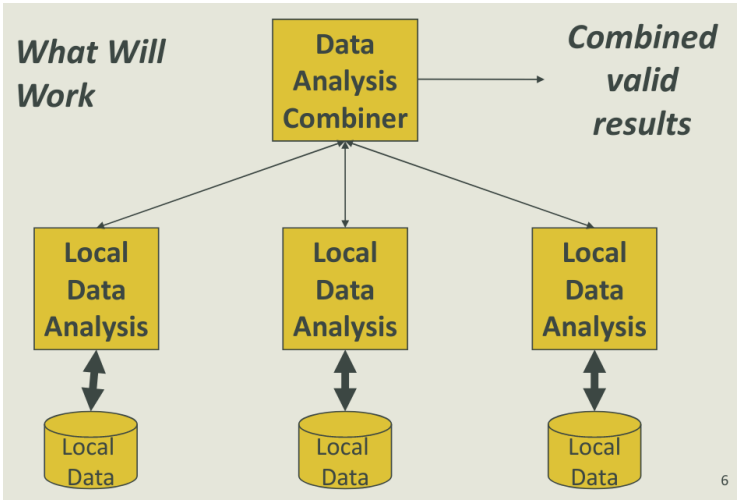
- **privacy**: i dati che non volevano essere condivisi tra i nodi si trovano tutti insieme;
- **performance**: tutto il carico dell'analisi dei dati viene riversato al data warehouse;
- **connettività**: ci sono grussi flussi di dati che devono circolare tra nodi locali e nodo centrale;
- **fonti eterogenee**: i nodi locali possono raccogliere dati di tipo diverso o chiamati/strutturati in maniera diversa: il processo di unificazione nell'unico nodo centrale potrebbe essere molto costoso.

## Soluzione al data warehouse

Possiamo pensare di inserire un modulo di analisi locale dei dati per ogni nodo in modo da partizionare il carico complessivo di dati da analizzare. Al




termine dell'analisi i risultati vengono comunicati ad un nodo che li aggrega e combina i risultati. Abbiamo quindi anche risolto il problema di privacy in quanto i dati reali non devono essere mai scambiati con nessuno, vengono infatti scambiati solamente i risultati intermedi dell'analisi, che saranno aggregati dal **data analysis combiner**. Le fonti eterogenee non vengono mai fuse tra di loro e il traffico è limitato ai soli risultati.




Spostiamo quindi il focus sul come effettuare localmente le analisi. Nelle grandi realtà le soluzioni adottate nei sistemi distributi per garantire confidenzialità e privacy sono:

- **Data Obfuscation**: i dati vengono resi rumorosi ad esempio con la Differential Privacy, nascosti dai nodi locali e quindi il modulo di analisi

che interagisce con loro non può mai vedere i dati reali ma solo loro versioni sintetiche o offuscate;
- **Data Summarization**: si estraggono delle sinossi (statistiche a vario livello) o riassunti come clustering e solo questi risultati vengono condivisi al modulo di analisi;
- **Data Separation**: dati rimangono nei nodi fidati e l'analisi viene fatta in maniera distribuita.

# Data Separation

I dati vengono mantenuti solo dal curatore o dal possessore e i processi di analisi vengono fatte o localmente o da terze parti fidate. I problemi di questo approccio risiedono nel capire sia se le terze parti siano o meno nelle condizioni di poter fare le analisi sia se i dati a seguito dell'analisi siano effettivamente privati. Gli approcci per garantire questi goals sono due: secure multiparty computation e federated learning.

## Secure Multiparty Computation

Molti degli approcci presentati per il multiparty si sono dimostrati molto buoni quando le parti coinvolte sono solo 2 anche perchè all'aumentare di queste aumenta sia la complessità algoritmica che l'overhead dell'intero processo. L'idea è calcolare una funzione sugli input che provengono da diversi nodi in cui nessuno vuole rivelare i dati agli altri nodi.

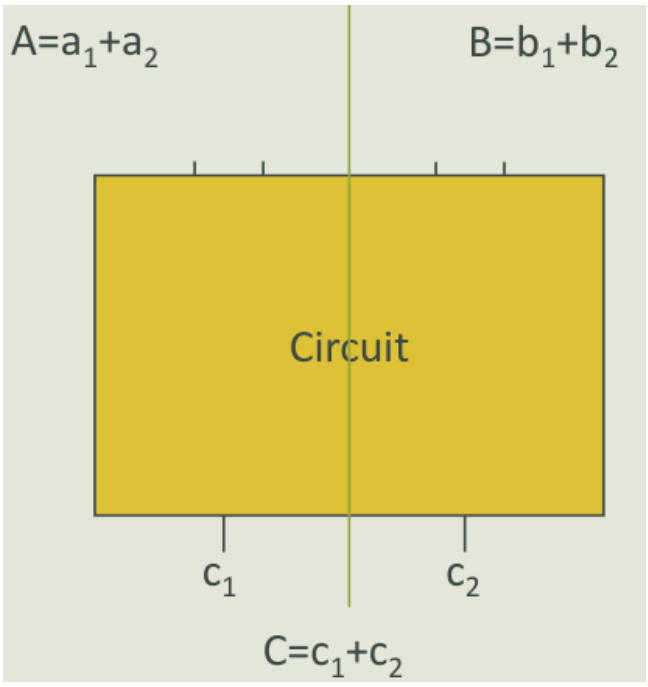
### Yao's Millionaire's problem

Problema noto che ha dato origine alla secure multiparty computation: ci sono 2 milionari che discutono tra di loro per capire chi tra i due è il più ricco, ma nessuno dei due vuole svelare la sua effettiva ricchezza. Vogliono trovare metodo e protocollo di comunicazione per portare a termine questo obiettivo.

Il problema è stato risolto rappresentando questa funzione che calcola il più ricco tra i due mediante un circuito dove le varie porte logiche che lo compongono calcolano il risultato in modo sicuro e privato. Questo approccio può essere generalizzato ad $n$ milionari, ma si complica parecchio.

L'idea è separare il circuito in due parti non comunicanti, ognuna acceduta da un milionario che passerrà le proprie informazioni al circuito. Alla fine viene eseguita una funzione sui due risultati:

- ogni parte ha la propria informazione in bit (supponiamo solo 2 bit a testa), aggiunge rumore ad una parte e la scambia con l'altro attore;
- ognuna delle parti da in input al circuito parte della sua informazione e parte ottenuta dall'altro attore;
- il circuito è composto da due XOR che processano in parallelo le due parti separatamente;




- il risultato delle due parti prima di essere sommato deve essere scambiato per fare in modo che entrambe le parti abbiano tutta l'informazione necessaria. Se la porta logica fosse un AND allora gli addendi dell'operazione finale dipenderebbero dai risultati ottenuti e dal protocollo di interazione, mentre per lo XOR l'operazione finale da compiere è sempre e solo su $c_1$ e $c_2$.

Questo è la modellazione del problema, la comunicazione che avviene sia al passo 1 sia per ottenere l'ouptut del proprio circuito viene invece risolta con la crittografia ed in particolare con il protocollo di **oblivious transfer** come strumento sicuro per la comunicazione tra le parti.

**Oblivious Transfer** Date due parti $A$ possiede i dati e $B$ fa le scelte. $A$ non sa le scelte che vengono fatte da $B$ e $B$ conosce solo i dati che sceglie. Al termine del protocollo $B$ conosce sono l'informazione di $A$ da lui scelta, mentre $A$ non impara nulla. Andrà fatto quindi anche a parti scambiate. Il protocollo funziona così:

1. $A$ invia la sua chiave pubblica a $B$;
2. $B$ sceglie quattro valori casuali $b_1, b_2, b_3, b_4$ ma solo uno di questi viene cifrato con la chiave di $A$, poi manda tutto ad $A$;
3. $A$ cerca di decifrare tutti i valori con la sua chiave privata e manda a $B$ i risultati di uno XOR tra i valori decifrati e il suo originario che possiede;

4. $B$ tra tutti i valori che riceve seleziona solo quello corrispondente a quello che ha scelto e da come output il risultato dello XOR tra questo e il valore che possiede. In questo modo $B$ conosce l'informazione che $A$ ha deciso di condivergli.

### Costruzione di albero decisionale

Vogliamo costruire un albero decisionale con la secure 2-party computation: lo schema del db è condiviso tra le parti, che mantengono però ovviamente privati i propri records e non vogliono rivelarli nel processo di costruzione del modello. Per costruire l'albero decisionale si userà l'algoritmo ID3 che essenzialmente sceglie per i nodi interni di split l'attributo che massimizza l'information gain o che, in maniera equivalente, minimizza l'entropia sulla divisione delle classi all'interno del dataset.

**Assunzioni e Limitazioni** Per estendere l'algoritmo ID3 facciamo le seguenti assunzioni/limitazioni:

- **modello semi-onesto**: il protocollo viene seguito in maniera fedele, ma dobbiamo tenere traccia di alcune computazioni intermedie che potrebbero essere problematiche se condivise;
- **solo 2 parti**: l'estensione ad un numero di parti maggiore non è per nulla banale;
- **algoritmo approssimato**: non calcoliamo l'originale ID3, ma una sua $\delta$ approssimazione: $\text{ID3}_\delta$. Il valore di $\delta$ incide sia sull'accuracy finale del modello sia sulla complessità del processo di costruzione;
- **attributi categorici**: funziona solo con attributi categorici.

Definiamo con $A$ l'insieme degli attributi, che possono essere usati una sola volta come nodo interno nella costruzione dell'albero, $C$ è l'attributo di classe e $D$ è l'insieme delle istanze. L'idea dell'algoritmo è selezionare come nodo interno ad ogni passo l'attributo che massimizza l'information gain e creare un nodo foglia quando tutte le istanze che ricadono in un segmento sono omogenee riguardo la classe. Continua così fino a quando non ha usato tutti gli attributi o fino a quando non ho istanze di una sola classe. Vediamo come rendere privati i diversi step che compongono l'algoritmo:

- **step 1**: se $A$ è vuoto va restituito un nodo foglia con la classe di maggioranza tra le istanze e il conteggio va fatto in maniera privata. Nello specificio si esegue il protocollo di Yao per sapere quale sia la classe di maggioranza: le due parti con due diversi dataset si condividono l'informazione sul numero di tuple per classe che possiedono;
- **step 2**: se $D$ ha istanze con stessa classe $c_i$ dobbiamo creare un nodo foglia etichettato con $c_i$. I nodi foglia con un numero di classi maggiore vengono rappresentati da una classe fissa $c_j \neq c_i$. A questo punto il protocollo viene usato per vedere se tutte le tuple abbiano o meno la stessa classe;

- **step 3**: per trovare l'attributo che garantisce il miglior information gain dobbiamo calcolare la funzione accedendo ai dati in $D$. Per fare il calcolo di $x \cdot \log(x)$ alla base dell'entropia in maniera privata si usa la sua approssimazione con il polinomio di Taylor che approssima la funzione come una serie di somme di monomi. Più termini utilizzo più ho approssimazione precisa, ma più overhead introduco.

### Federated Learning

La motivazione dietro alla nascita del **federated learning** da parte di Google è dovuta al sempre crescente problema di accesso ai dati. I singoli nodi non vogliono condividere i dati che possiedono, ma permettono che questi vengano usati localmente per il learning. Dall'altra parte c'è la necessità per il deep learning di tanti dati per fare un apprendimento sensato. Da qui nasce l'esigenza di fare un learning distribuito e collaborativo che sia però rispettoso della privacy.

## Federated Learning Centralizzato

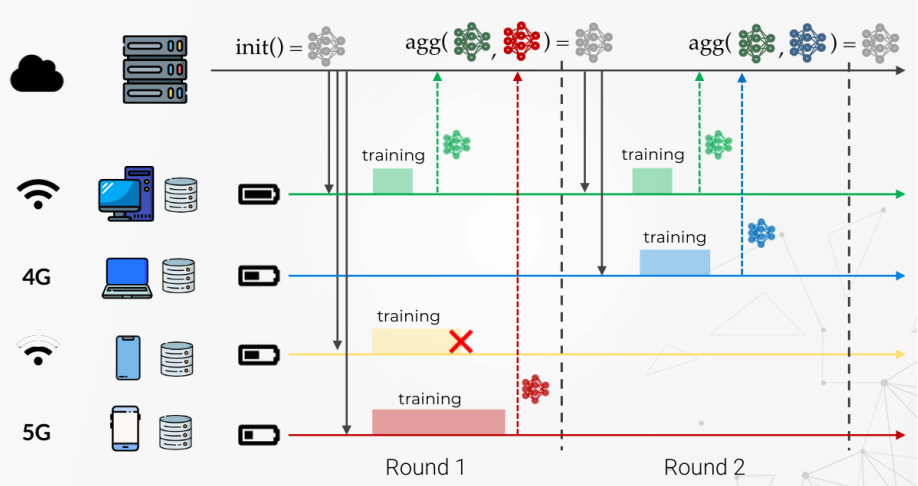
Si tratta di un setting di machine learning in cui più entità detti **clients o nodi** collaborano per risolvere un problema di machine learning coordinati da un **server centrale** o **aggregatore**. I dati dei clients, solitamente esigui datasets, non vengono scambiati tra le parti ed il learning è focalizzato sull' aggiornamento (updates locali) del modello ed aggregazione dei vari risultati nel server centrale. La forza di questo meccanismo è la scalabilità ad un numero di nodi mostruosamente grande: posso andare da scenario con meno di 100 nodi con datasets medio-grandi (*cross-silo federated learning*), fino a milioni di nodi con datasets minuscoli (*cross-device federated learning* scenario di Google). Il federated learning può anche essere:

- **orizzontale**: se ogni utente ha un sottoinsieme degli esempi totali della tabella immaginaria con tutte le colonne, ossia ogni utente ha dei records;
- **verticale**: se ogni utente ha tutte le informazioni di riga, ma per pochi attributi.

Il federated learning fa assunzione che i parametri del modello, oggetto degli scambi tra nodi e server, non contengano informazioni riguardo ai dati usati per il training e che la dimensione del modello scambiato sia più piccola dei dati di training. I goals del federated learning è garantire privacy e ottenere un modello che sia il più simile possibile a quello che avrebbe appreso se avesse centralizzato i dati. Un round di training è caratterizzato dai seguenti steps:

1. Il server centrale inizializza il modello con parametri casuali per il task che deve risolvere;
2. Condivide il modello con tutti i nodi che localmente lo aggiornano con i propri dati privati;
3. Ogni nodo restituisce il modello al server centrale che aggrega e genera il nuovo modello globale. Il processo si ripete fino ad un certo criterio di

convergenza.




Come possiamo vedere dall'immagine il modello inizializzato nel server viene inviato anche solo ad un sottoinsieme dei nodi e non è richiesto che tutti riescano a portare a buon fine il processo di training. Ogni round può essere visto come una classica epoca del deep learning standard. In quest'ottica ogni nodo è uno dei batch che compone l'epoca.

## Challenges

Quali sono le sfide che il federated learning deve affrontare:

- ogni nodo può avere dei dati locali con distribuzione completamente diversa da quella della popolazione totale e questi dati **non** indipendenti e **non** identicamente distribuiti possono causare dei problemi in fase di apprendimento;
- ci può poi essere un carico completamente sbilanciato con utenti con molti più dati di altri;
- è distribuito in maniera massiva ossia il numero potenziale di utenti è molto maggiore del numero medio di esempi mantenuti da ognuno di questi;
- la rete di comunicazione può essere altamente instabile tra il server centrale e ogni singolo nodo.

## Federated SGD

Il **federated SGD** è l'approccio più semplice per fare federated learning. In questo contesto un client selezionato in maniera randomica è il corrispondente all'incirca di un batch di un approccio standard di deep learning. Il federated SGD esegue ad ogni round l'aggiornamento del modello su un sottoinsieme dei clients che compongono la rete scelti in maniera randomica/stocastica, classico dell'approccio SGD. Ci sono anche dei criteri parzialmente deterministici per

la scelta di un client come il fatto che sia un dispositivo recente (e quindi sufficientemente potente), che sia in carica, ecc... Se prendiamo tutti i nodi della rete per un singolo round siamo nell'approccio full batch e non è realistico dato il numero totale di dispositivi che partecipano. Nell'approccio SGD abbiamo due alternative:

1. ogni client calcola solamente il gradiente della funzione di loss sul modello e lo manda al server che farà l'aggioranamento del modello usando unico vettore che è media pesata dei gradienti dei clients. Il peso dipende dalla frazione del numero di esempi totali che un utente possiede;
2. ogni client calcola il gradiente ed aggiorna i pesi del proprio modello che viene rimandato al server che dovrà ancora aggregare i modelli facendone una media pesata.

A livello matematico dell'apprendimento non cambia nulla, ma il secondo approccio lo si può generalizzare ed evitare di mandare continuamente informazioni al server ogni volta che vi è un calcolo di un gradiente che è altamente inefficiente dal punto di vista della comunicazione. Si può ancora migliorare o selezionando un numero maggiore di utenti per round in modo da ottenere risultati più accurati e avere meno rounds totali oppure facendo fare più computazioni ad ogni clients, più epoche. Questo ultimo approccio prende il nome di **FedAVG** che, nonostante funzioni bene in pratica, non garantisce convergenza lineare nemmeno per funzioni di loss convesse se il numero di epoche è maggiore di 1 e questo è dovuto alle challenges prima identificate come la presenza di dati non i.i.d.

## Federated Learning Decentralizzato

Non vi è più la figura del server centrale e tutto avviene con comunicazione tra clients. Uno dei modi per fare questo è il **gossip learning** la cui idea è che ogni nodo inizializza il modello in maniera causale localmente e a cadenza regolare sceglie un nodo peer a cui inviare il proprio modello per poi rimanere in attesa di ricevere modelli da altri nodi. Appena ne riceve uno fa una merge e update con il proprio e ripete il tutto. Il gossip learning funziona bene a livello pratico.

## Privacy nel FL

L'unica garanzia di privacy che promuove il federated learning è quella di addestrare il modello in maniera distribuita senza che i nodi possano in alcun modo conoscere i dati usati da altri nodi della stessa rete. Il problema è che questo non è del tutto vero, è possibile per un nodo ricostruire parte dei dati di training specialmente per modelli fully connected, per i quali esisteno anche teoremi a supporto.

## Possibili Attacchi

Quanto detto prima fa si che ci siano due tipi di attaccanti:

- **semi-onesto**: l'attaccante partecipa in maniera onesta al federated learning, ma ogni qual volta riceve un modello ne può fare un analisi per cercare di estrarre informazioni su altri nodi. Per un clients questo è difficile perchè riceve sempre un modello aggregato, non è invece così per il server che riceve il modello addestrato su singoli dati degli utenti e può fare delle inferenze molto mirate;
- **malizioso**: oltre al precedente comportamento può anche partecipare in maniera non corretta al protocollo inviando al server non il modello che ottiene a seguito dell'addestramento ma un modello ad hoc che gli permetta nei round successivi di trarre maggiori informazioni sui nodi.

Ci sono due possibili attacchi per ottenere informazioni sui dati di training:

- **membership inference**: dati degli esempi si vuole capire se sono stati usati per fare il training;
- **model inversion**: imparare quali sono i dati che sono stati usati per fare il training.

Questi possono essere portati a termine o usando il modello condiviso come una black-box e approccio esaustivo oppure usando modelli di ML appositamente costruiti per fare inferenza sul valore dei pesi o del gradiente ottenuto.

## Meccanismi di Difesa

### Homomoriphic Encryption

L'idea alla base è fare operazioni matematiche su dati criptati per ottenere la versione criptata del risultato dell'operazione matematica che avrei potuto fare sui dati in chiaro. Questo vale generalmente solo per operazioni di somma e moltiplicazione. L'idea è che i clients criptino i modelli che devono inviare al server al fine del round, il quale aggregandoli otterrà lo stesso modello cifrato che avrebbe ottenuto facendo prima l'aggregazione e poi la cifratura. Reinvia il modello ai clients che possono decifrarlo, farne l'update, cifrarlo e reinviarlo. Il problema di questo approccio è che è molto dispendioso dal punto di vista computazionale per i nodi che spesso sono solo smartphones.

### Differential Privacy

Si può usare la differential privacy in due modi diversi:

- **locale**: i singoli clients prima di condividere il modello al server applicano del rumore. Il problema è che fare ogni volta aggiornamento e distorcerlo con del rumore rende il learning molto meno efficace;
- **globale**: il rumore è aggiunto dal server dopo l'aggragazione dei modelli, alla fine di ogni round.

**Secure Aggregation**

Fa parte di una classe di algoritmi di Secure Multiparty Aggregation in cui si vuole calcolare una funzione aggregata su più parti che però non si fidano tra di loro. Il modo di procedere è simile alla homomorphic encryption ovvero ogni clients invia il modello con una certa maschera che è stata scelta in accordo con le altre parti che partecipano all'aggregazione, in modo tale che il server quando li aggrega ottiene come risultato implicito ed indiretto l'eliminazione delle maschere a vicenda.