fa

Università degli Studi di Salerno

.DIEM

Dipartimento di Ingegneria dell'Informazione ed Elettrica e Matematica Applicata

Corso di Laurea in Ingegneria Informatica

Basi di Dati 2021/2022
Canale I-Z

Project Work

Traccia N. 7 - Emergenza Covid

Gruppo n. **67 – IZ**

WP	Cognome e Nome	Matricola	e-mail	Responsabile
1	Intonti Giovanni	0612704899	g.intonti@studenti.unisa.it	X
2	Venditti Francesca	0612704699	f.venditti1@studenti.unisa.it	

3	Tarantino Ramona	0612704902	r.tarantino7@gmail.com	
4	Milone Marco	0612704869	m.milone15@gmail.com	

Anno accademico 2021-2022

Sommario

1. Descrizione della realtà di interesse

Titolo: **Emergenza Covid**

Al momento, in Italia, circa le informazioni relative all'epidemia di Covid, esistono diverse fonti che offrono informazioni e dati, peraltro organizzate in modo differente. In particolare sul sito https://it.wikipedia.org/wiki/Gestione_della_pandemia_di_COVID-19_in_Italia è reperibile lo storico della colorazione delle regioni, con riferimenti alle ordinanze governative e altri dettagli; sul sito: https://www.salute.gov.it/portale/nuovocoronavirus/archivioNotizieNuovoCoronavirus.jsp il quadro sintetico degli indicatori settimanali del Ministero della Salute, Istituto Superiore di Sanità, Cabina di Regia che riporta settimanalmente e per regione i nuovi casi segnalati nella settimana; trend settimanale COVID19; stima di RT-puntuale; dichiarata trasmissione non gestibile in modo efficace con misure locali (zone rosse); valutazione della probabilità; valutazione di impatto; allerte relative alla resilienza dei servizi sanitari territoriali; compatibilità RT sintomi puntuale con gli scenari di trasmissione; classificazione complessiva di rischio; classificazione alta e/o equiparata ad Alta per 3 o più settimane consecutive. Inoltre sono presenti i seguenti indicatori di cui al DM: Ind1.1 settimana precedente (%); Ind1.1 settimana di riferimento (%); Variazione; Ind1.2 (%); Ind1.3 (%); Ind1.4 (%); Ind2.1* (precedente); Ind2.1# (settimana di riferimento); Ind2.2 (mediana giorni tra inizio sintomi e diagnosi**); Ind2.3 (mediana); Ind2.4; Ind2.5; Totale risorse umane; Ind2.6; Resilienza dei servizi sanitari territoriali; Ind3.1; Trend 3.1 (% variazione settimanale); Trend 3.4 (% variazione settimanale); Ind3.2 (Rt puntuale); Ind3. 5; Ind3.6; Ind3.8*; Ind3.9*.

Infine sono presenti, su alcuni siti, dati puntuali giornalieri, suddivisi per regione, relativi a: data stato; codice_regione; denominazione_regione; lat; long; ricoverati_con_sintomi; terapia_intensiva; totale_ospedalizzati; isolamento_domiciliare; totale_positivi; variazione_totale_positivi; nuovi_positivi; dimessi_guariti; deceduti; casi_da_sospetto_diagnostico; casi_da_screening totale_casi; tamponi; casi_testati; note; ingressi_terapia_intensiva, ecc.

La base di dati dovrà memorizzare gli indicatori precedentemente indicati (è possibile specializzare il database anche rispetto ad una sola categoria di indicatori, e.g., resilienza del sistema sanitario; dati epidemiologici; etc, come indicato nel seguito), e eventualmente ulteriori indicatori identificati in dataset disponibili. Esistono infatti diversi indicatori (si faccia riferimento al decreto https://www.trovanorme.salute.gov.it/norme/renderNormsanPdf?anno=2020&codLeg=77099&p arte=1%2 0&serie=null e ai report di monitoraggio settimanale quali http://www.salute.gov.it/imgs/C_17_monitoraggi_49_0_fileNazionale.pdf).

- Di ogni indicatore è di interesse conoscere il codice di riferimento, la categoria a cui appartiene, una descrizione,
- Un indicatore per essere calcolato ha bisogno di dati/attributi (ad esempio il tasso di posti letto occupati (indicatore) è calcolato come rapporto tra il numero di posti letto occupati e il numero di posti totali disponibili (attributi)). Questi dati devono essere memorizzati, insieme alla data in cui vengono rilevati e alla località cui fanno riferimento
- È necessario memorizzare i dati relativi ai diversi pronto soccorso: dove si trovano (indirizzo e se sono presso un ospedale specifico); grandezza; etc.
- Sono di interesse informazioni sul numero di accessi al pronto soccorso giornaliero; se l'accesso è per sospetto COVID; il tempo medio di accesso; etc.
- È di interesse memorizzare i dati per singola città, provincia, regione.

Possibili alternative: Fermo restando la rappresentazione di alcune informazioni rilevanti (e.g., indicatori, regioni, etc.), sulla base delle informazioni disponibili (quelle precedentemente indicate e quelle disponibili in altre fonti) può essere di interesse progettare e sviluppare, in alternativa:4

A) Una base di dati che consenta di memorizzare i dati relativi agli indicatori sintetici settimanali al fine di effettuare ulteriori analisi;

B) Una base di dati che consenta di memorizzare i dati giornalieri (in ultima analisi anche settimanali) di base da cui ricavare gli indicatori utilizzati e/o altri, a partire da informazioni disponibili in rete nei diversi dataset.

C) Una base di dati che consenta di organizzare e strutturare le informazioni presenti sul sito di Wikipedia precedentemente indicato, con particolare riferimento alle colorazioni delle regioni; in particolare, indicare la regione, la settimana di riferimento, il colore imposto e il decreto da cui viene determinato la decisione del particolare colore, settimane di riferimento considerate per determinare la colorazione ed alcuni dei principali dati di resilienza, infezione, ecc.

D) Una base di dati a livello di ospedale o ASL – anche a livello comunale - in grado di memorizzare dati e informazioni per poi poter costruire gli indicatori di utilità in analisi di rischio. Quest' ultima base di dati può trattare ovviamente l'informazione a livello di singolo paziente e/o caso e/o Cittadino.

1.1. Analisi della realtà di interesse

Lo scopo del progetto sarà quello di realizzare una base di dati in grado di memorizzare informazioni relative al COVID-19, trattato dal punto di vista dei casi soffermandoci in particolar modo sull'analisi di rischio e sul monitoraggio del virus nell'ambito universitario. Le università prese in considerazione riguarderanno quelle situate in territorio italiano, inoltre un singolo studente deve essere iscritto ad una sola università. Il progetto verterà sulla memorizzazione di dati relativi agli studenti, i loro ingressi alla struttura, i loro contatti all'interno dell'Università. Un obiettivo principale del database sarà, inoltre, il monitoraggio dei positivi e il relativo tracciamento, obiettivo conseguito grazie alla memorizzazione dei contatti avuti tra i vari studenti e alla memorizzazione delle positività degli stessi, prestando particolare attenzione al giorno di avvenuto contatto e alla situazione medica dello studente. Inoltre, l'idea è quella di fornire uno storico dettagliato riguardante i casi covid rilevati.
Abbiamo deciso di non soffermarci nell'ambito ospedaliero perché trattando dati relativi a studenti la cui età varia trai i 18 e i 28 anni siamo consci del fatto che una percentuale minimale di essi è costretta ad un ricovero. Per quanto riguarda gli indicatori regionali, abbiamo deciso di non trattarli in senso stretto, ma di condurre statistiche in base alla regione di appartenenza dell'Università per ricavare informazioni.

2. Analisi delle specifiche

Workpackage	***Task***	***Responsabile***
WP0	Analisi delle specifiche	Intero Gruppo

2.1. Glossario dei termini

	Termine	Descrizione	Sinonimi	Collegamenti
1	Università	Istituzione didattica per l'istruzione ed il rilascio di titoli accademici.	-	Persona, Aula
2	Persona	Lo studente iscritto all'università o il professore che afferisce ad essa.	-	Università, Ingresso
3	Caso COVID	Persona che viene infettata dal virus Covid-19 o entra in contatto con un positivo durante un'attività.	-	Persona, Tampone
4	Ingresso	L'ingresso alla struttura, in particolare all'aula in cui viene svolta un'attività da un professore e a cui gli studenti partecipano	-	Persona ,Attività
5	Tampone	Tampone, ovvero un test, per documentare la positività o negatività al Covid-19 effettuato da un caso covid.	-	Caso Covid
6	Aula	Aula dell'Università dove vengono erogate le attività didattiche.	-	Università, Attività
7	Attività	Attività intesa come lezione, meeting, incontri aperti agli studenti con numero massimo di posti stabilito dall'effettiva capienza dell'aula e dalle norme anti-Covid.	-	Aula, Ingresso

Tabella 1. Glossario dei Termini

2.2. Strutturazione dei requisiti in frasi

2.2.1. Frasi di carattere generale

Si vuole realizzare una base di dati per la memorizzazione di dati e informazioni relative al COVID-19 in ambito universitario e a livello di singolo studente o professore.

2.2.2. Frasi relative a Università

Per L'Università rappresentiamo indirizzo, città, nome della struttura, regione in cui è collocata e infine i posti disponibili all'interno della stessa. Per essere definita tale, l'Università ha bisogno di almeno uno studente iscritto e almeno un professore che afferisca alla struttura. L'università, inoltre, può esistere se ha almeno un'aula.

2.2.3. Frasi relative a Persona

Relativamente alle persone rappresentiamo il codice fiscale, il nome, il cognome, la matricola, il sesso, la data e la città di nascita, e-mail, il dipartimento ed il loro stato attuale. La persona deve afferire ad una ed una sola università. La persona può essere uno studente o un professore.

2.2.4. Frasi relative a tipi specifici di Persona

Per gli studenti rappresentiamo l'anno di corso, il loro curriculum e l'anno di iscrizione all'Università. Oltre agli studenti, rappresentiamo anche i professori, di cui rappresentiamo il titolo e il numero di ufficio. Più professori possono avere lo stesso ufficio.

2.2.5. Frasi relative a Caso COVID

Per il caso covid rappresentiamo l'ID del caso covid e il vaccino se è stato effettuato. Una stessa persona potrebbe aver effettuato diversi vaccini. Una persona che viene infettata dal virus, o entra in contatto con un positivo, potrebbe riesserlo più volte nel corso della permanenza all'università. Per il caso covid vogliamo memorizzare i tamponi effettuati per tracciare la sua positività/negatività.

2.2.6. Frasi relative a tipi specifici di Caso COVID

Per i casi covid derivanti da una positività, specifichiamo anche se è stato ospedalizzato, se ha mostrato sintomi, oltre alla data di positività e di negatività. Inoltre, esso avrà bisogno del tampone di riferimento della positività. Nel caso in cui il caso covid non derivi da una positività ma da un contatto deve essere specificata la data in cui esso è avvenuto. Un caso covid da contatto deve effettuare un tampone per modificare il suo stato in seguito ad un periodo di quarantena che dipende dal suo vaccino.

2.2.7. Frasi relative a Tampone

Per quanto riguarda il tampone rappresentiamo il codice CUN che lo identifica nazionalmente, l'orario e la data in cui è effettuato e il relativo esito. Consideriamo anche la tipologia del tampone. Il tampone deve riferirsi ad un caso covid positivo o ad uno da contatto, e solo ad uno dei due.

2.2.8. Frasi relative ad Aula

Per l'aula rappresentiamo il nome, l'edificio, il numero di studenti che può contenere.

2.2.9. Frasi relative a Attività

Per le attività rappresentiamo l'orario di inizio, l'orario di fine, la massima capienza stabilita dalle norme covid e la relativa modalità. Viene considerato anche il giorno della settimana in cui verrà effettuata. Non possono esserci contemporaneamente due o più attività nello stesso momento nella stessa aula.

2.2.10. Frasi relative a Ingresso

Per gli ingressi rappresentiamo la persona che effettua l'ingresso , l'attività alla quale partecipa e la data in cui effettua l'ingresso.

2.3. Identificazione delle operazioni principali

Operazione 1: Per ogni università, stampo storico caso covid di una persona (280 volte a settimana)

Operazione 2: Per ogni università, aggiornamento caso covid (negativizzazione) (7 volte a settimana)

Operazione 3: Ricerca aule libere in uno slot orario (14 volte a settimana)

Operazione 4: Per ogni università, ricerca delle attività del giorno svolte dai positivi odierni (7 volte a settimana)

Operazione 5: Per ogni università, tracciamento dei contatti (7 volte a settimana)

Operazione 6: Statistica per università (1 volta a settimana)

Operazione 7: Per ogni università, ricerca positivi da più di 21 giorni (7 volte a settimana)

3. Progettazione Concettuale

Workpackage	***Task***	***Responsabile***
WP1	Progettazione Concettuale	Intonti Giovanni

3.1. Schema Concettuale

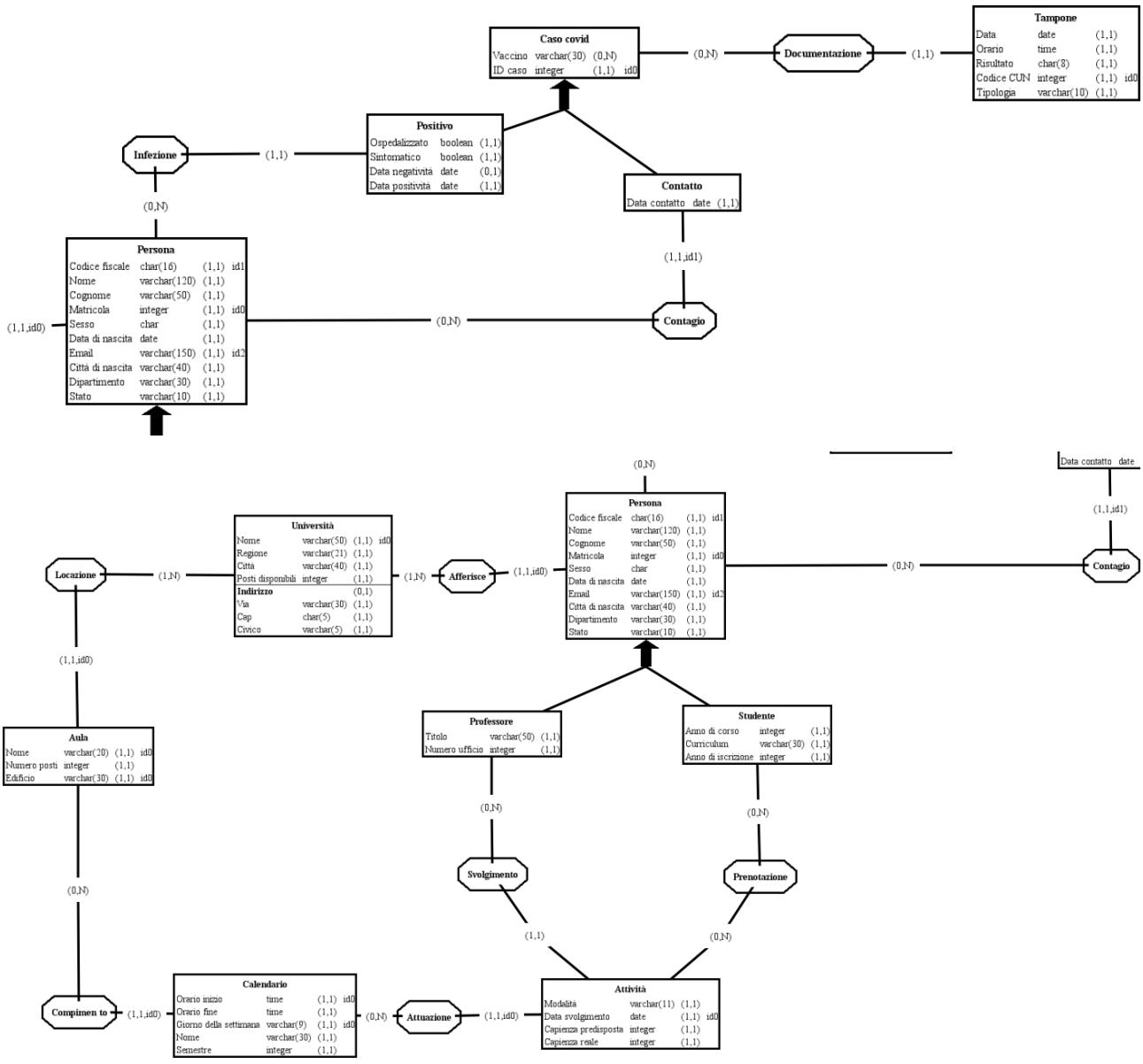


Figura 1. Schema E-R

3.1.1. Note sullo schema E-R *[opzionale]*

La generalizzazione Persona -> (Professore, Studente) è totale perché nella nostra realtà di interesse consideriamo partecipanti attivi all'università soltanto queste due entità.

3.2. Design Pattern

3.2.1. Pattern Part Of

Il pattern "Part Of" è stato applicato poiché l'entità Aula è parte dell'entità Università, l'esistenza dell'occorrenza dell'entità Aula dipende dall'esistenza di un'occorrenza dell'entità di Università e richiede l'identificazione esterna, in questo caso Nome dell'Università.

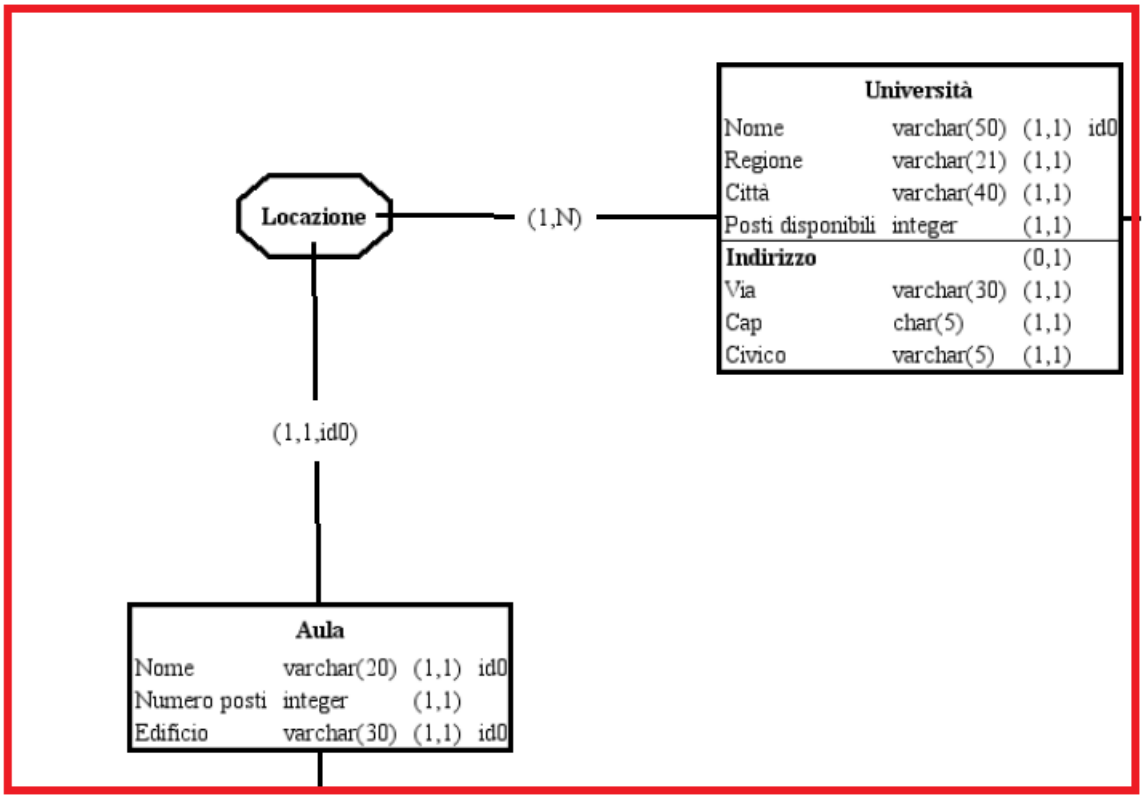


Figura 2 Schema successiva all'applicazione del Pattern Part Of. (opzionale)

3.2.2. Pattern Instance Of

Il pattern "Instance Of" è stato applicato poiché è presente un'entità che rappresenta il concetto astratto di attività programmata, Calendario, caratterizzata da Orario inizio, Orario fine, Giorno della settimana, Nome e Semestre, inoltre è presente l'entità Attività che rappresenta la particolare istanza del calendario in una certa data. L'identificazione dell'attività avviene attraverso la data ed esternamente l'attività programmata di cui è istanza.

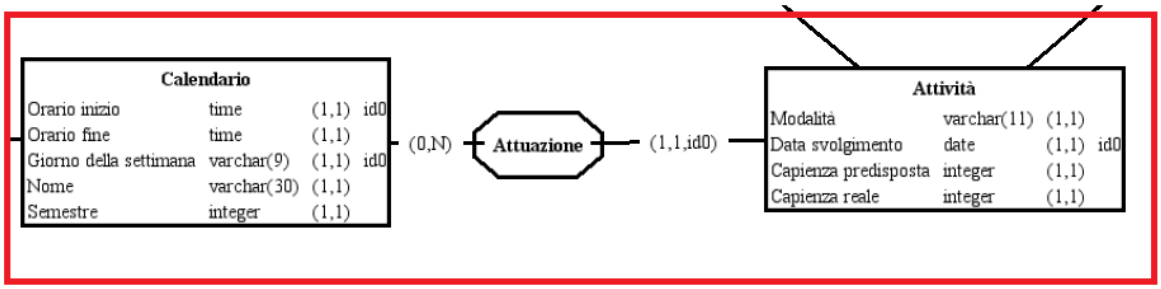


Figura 3 Schema successiva all'applicazione del Pattern Istance Of. (opzionale)

3.3. Dizionario dei Dati

[Inserire qui il Dizionario dei Dati]

Entità	Descrizione	Attributi	Identificatore
Università	Luogo in cui vengono schedulate attività a cui partecipano gli studenti e svolte dai professori	Nome, Regione, Città, Posti disponibili, Indirizzo	Nome
Persona	Persona che frequenta l'università	Codice fiscale, Matricola, Nome, Cognome, Sesso, Data di nascita, E-mail, Città di nascita, Dipartimento, Stato	Codice fiscale (1) Matricola, Nome(Università) (2) E-mail (3)
Studente	Persona iscritta all'università (Generalizzazione di Persona)	Anno di corso, Curriculum, Anno di iscrizione	Come l'entità Padre
Professore	Persona che svolge attività all'interno dell'università (Generalizzazione di Persona)	Titolo, Numero ufficio	Come l'entità Padre
Attività	Attività didattica a cui gli studenti possono accedere e svolte dai professori in una data precisa.	Modalità, Data svolgimento, Capienza predisposta, Capienza reale	Data svolgimento, Orario inizio(Calendario), Giorno della settimana (Calendario), Nome(Aula), Edificio (Aula), Nome(Università)
Aula	Luogo in cui vengono svolte le attività	Nome, Edificio, Numero Posti	Nome, Edificio, Nome (Università)
Calendario	Attività didattiche schedulate durante la settimana in un relativo semestre	Orario inizio, Orario fine, Giorno della settimana, Nome, Semestre	Orario inizio, Giorno della settimana, Nome (Aula), Edificio (Aula), Nome (Università)
Tampone	Test effettuato per attestare positività o negatività al Covid-19	Data, Orario, Risultato, Codice CUN, Tipologia	Codice CUN
Caso Covid	Persona che risulta coinvolta in	Vaccino, ID caso	ID Caso

	un'infezione da covid-19		
Positivo	Persona che risulta positiva al virus in seguito ad un tampone che ne attesti la positività (Generalizzazione di Caso Covid)	Ospedalizzato, Sintomatico, Data negatività, Data positività	Come l'entità Padre
Contatto	Persona che entra in contatto, tramite un'attività, con un'altra persona che ha contratto il virus	Data contatto	Come l'entità Padre

Tabella 2. Dizionario dei dati – Entità

Relazioni	Descrizione	Entità Coinvolte	Attributi
Afferisce	Lega la persona all'università a cui è iscritta	Persona, Università	
Locazione	Lega l'aula all'università a cui appartiene	Università, Aula	
Compimento	Lega l'attività del calendario all'aula in cui è svolta	Aula, Calendario	
Attuazione	Lega l'attività schedulata in un giorno particolare alla sua programmazione nel calendario	Calendario, Attività	
Svolgimento	Lega il professore alle attività che svolge	Professore, Attività	
Prenotazione	Lega lo studente alle attività a cui partecipa	Studente, Attività	
Infezione	Lega il caso covid positivo alla persona a cui fa riferimento	Persona, Positivo	
Contagio	Lega il caso covid da contatto alla persona a cui fa riferimento	Persona, Contatto	
Documentazione	Lega il caso covid ai tamponi effettuati	Caso covid, Tampone	

Tabella 3. Dizionario dei dati - Relazioni

Workpackage	***Task***	***Responsabile***
WP4	Regole Aziendali	Milone Marco

3.4. Regole Aziendali

Regole di Vincolo
(RV1) L'attività deve essere svolta in modalità mista nel caso di un numero di prenotazioni superiore alla capienza predisposta. **(RV2)** L'attività deve essere svolta in modalità mista nel caso di uno studente positivo al covid-19 prenotato alla medesima. **(RV3)** L'attività deve essere svolta in modalità a distanza nel caso di positività del professore che svolge la medesima. **(RV4)** Il caso covid da contatto vaccinato con 3 dosi deve effettuare un tampone dopo almeno 7 giorni dalla data di contatto (quarantena). **(RV5)** Il caso covid da contatto vaccinato con 2 dosi da meno di 120 giorni deve effettuare un tampone dopo almeno 7 giorni dalla data di contatto(quarantena). **(RV6)** Il caso covid da contatto non vaccinato deve effettuare un tampone almeno dopo 10 giorni dalla data di contatto. (quarantena). **(RV7)** Il caso covid da contatto vaccinato con 1 dose deve effettuare un tampone almeno dopo 10 giorni dalla data di contatto. (quarantena). (RV8) Il caso covid da contatto con due dosi da più di 120 giorni deve effettuare un tampone almeno dopo 10 giorni dalla data di contatto. (quarantena). **(RV8)** I casi covid e i tamponi memorizzati sono relativi all'anno accademico in corso. **(RV9)** Un'attività è svolta da un solo professore. **(RV10)** Un positivo deve avere al massimo 1 tampone di riferimento positivo e 1 tampone di riferimento negativo. **(RV11)** Un contatto deve avere al massimo un tampone di riferimento. *Nota: consideriamo i vaccini monodose al pari di quelli con più dosi; quindi, avere una dose di Moderna significa averne soltanto una.

Tabella 4. Regole di vincolo

Regole di derivazione
(RD1) La capienza reale di un'attività coincide con il numero di posti dell'aula in cui è effettuata. **(RD2)** La data di negatività di un caso covid coincide con la data del tampone risultato negativo che ha inserito come documentazione. **(RD3)** Il numero di posti disponibili dell'università coincide con il numero di posti delle aule della stessa.

Tabella 5. Regole di derivazione

4. Progettazione Logica

Workpackage	***Task***	***Responsabile***
WP2	Progettazione Logica	Venditti Francesca

4.1. Ristrutturazione Schema Concettuale

4.1.1. Analisi delle Prestazioni

4.1.1.1. Tavola dei volumi

Concetto	Tipo	Volume
Persona	E	1520000
Studente	E	1500000
Università	E	100
Professore	E	20000
Calendario	E	150000
Attività	E	1500000
Aula	E	10000
Tampone	E	6000000
Caso Covid	E	4500000
Positivo	E	450000
Contatto	E	4050000
Afferisce	R	1520000
Locazione	R	10000
Compimento	R	150000
Attuazione	R	1500000
Svolgimento	R	1500000
Infezione	R	450000
Documentazione	R	6000000
Contagio	R	4050000

Tabella 6. Tavola dei volumi

I volumi scelti si basano su queste considerazioni:

- Le università italiane sono circa 100.
- Gli studenti universitari italiani sono 1500000, i professori 20000.
- In media un gruppo di 150 persone segue le stesse attività, dato che consideriamo 3 corsi a semestre con 2,5 lezioni a settimana otteniamo le attività del calendario.
- Dato che in media lo stesso gruppo di persone segue quelle attività per 10 settimane a semestre, otteniamo il numero di attività.
- Ogni università ha circa cento aule.
- Basandoci sulla nostra esperienza universitaria durante la pandemia, abbiamo considerato una media di 4 tamponi effettuati da ogni studente per attestare il suo inizio positività e fine, oppure la sua negatività in seguito ad un contatto. Otteniamo così il numero di tamponi.
- Basandoci sulla stessa media, dato che consideriamo casi covid anche i contatti con un positivo e considerando che non esiste una relazione uno a uno tra tampone e caso covid, dato che un caso covid positivo avrà una documentazione per la positività e uno per la negatività, otteniamo il numero di casi covid della nostra base dati.

4.1.1.2. Tavola delle operazioni

Operazione	Tipo	Frequenza
Operazione 1: Per università, storico caso covid di una persona	B	*280 volte a settimana*
Operazione 2: Per università, aggiornamento caso covid negativo	I	*7 volte a settimana*
Operazione 3: Per università, ricerca aule libere in uno slot	I	*14 volte a settimana*
Operazione 4: Per università, ricerca casi covid odierni e attività svolte da loro	I	*7 volte a settimana*
Operazione 5: Per università, tracciamento contatti	I	*7 volte a settimana*
Operazione 6: Statistica per università	B	1 volta a settimana
Operazione 7: Per università, ricerca positivi da più di 21 giorni e aggiornamento dello stato	I	*7 volte a settimana*

Tabella 7. Tavola delle operazioni

4.2. Analisi delle ridondanze

- *Ridondanza 1: Data negatività del caso covid. È ridondante poiché è derivabile dal tampone negativo effettuato dal caso covid, è un attributo derivabile da altre entità.*
- Ridondanza 2: Posti disponibili dell'Università. È ridondante poiché è derivabile dal conteggio del numero di aule presenti nell'università stessa.
- Ridondanza 3: Capienza reale dell'Attività. È ridondante poiché derivabile dal numero di posti presenti nell'aula in cui è schedulata l'attività.

Abbiamo deciso di analizzare la ridondanza 1 poiché è quella che è di maggiore interesse nella realtà di interesse, consci dell'aver riconosciuto anche le altre.
Nota: L'attributo data positività non lo abbiamo considerato ridondante poiché per la nostra realtà di interesse la positività ha inizio quando viene segnalata dalla persona che può caricare un tampone anche antecedente.

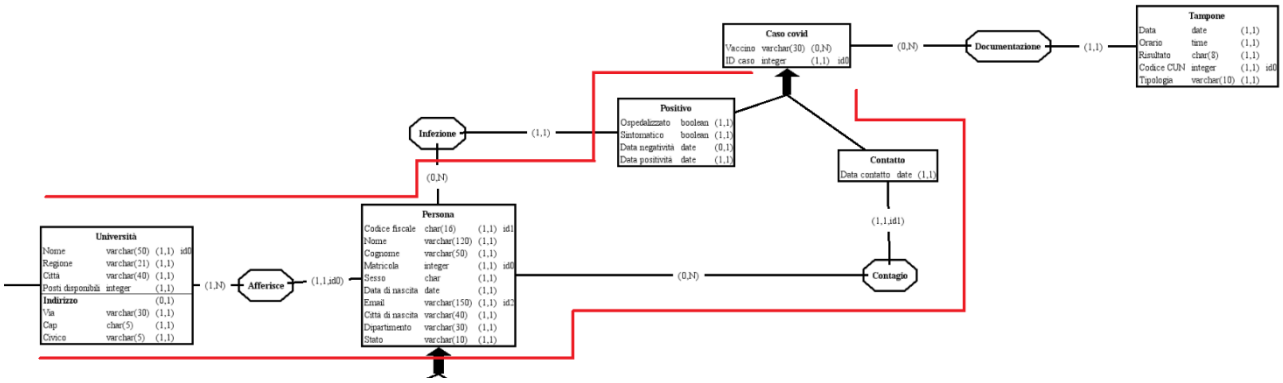
4.2.1. Analisi della ridondanza 1: Data negatività

- **Operazione 1:** Storico caso covid di una persona, per università

Con Ridondanza

CONCETTO	COSTRUTTO	ACCESSI	TIPO
UNIVERSITA'	E	1	L
AFFERISCE	R	1	L
PERSONA	E	1	L
INFEZIONE	R	1	L
POSITIVO	E	1	L

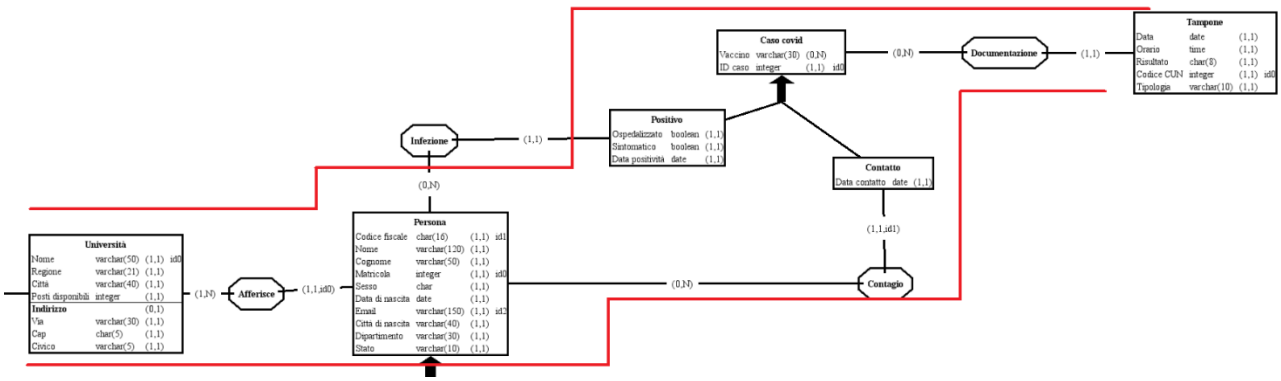
CASO COVID	E	1	L
UNIVERSITA’	E	1	L
AFFERISCE	R	1	L
PERSONA	E	1	L
CONTAGIO	R	10	L
CONTATTO	E	10	L
CASO COVID	E	10	L
DOCUMENTAZIONE	R	10	L
TAMPONE	E	10	L



Nota* = Abbiamo considerato, basandoci sulla tavola dei volumi, che una persona in media è positiva una volta l’anno e abbia 10 contatti con positivi durante l’anno.

Senza Ridondanza

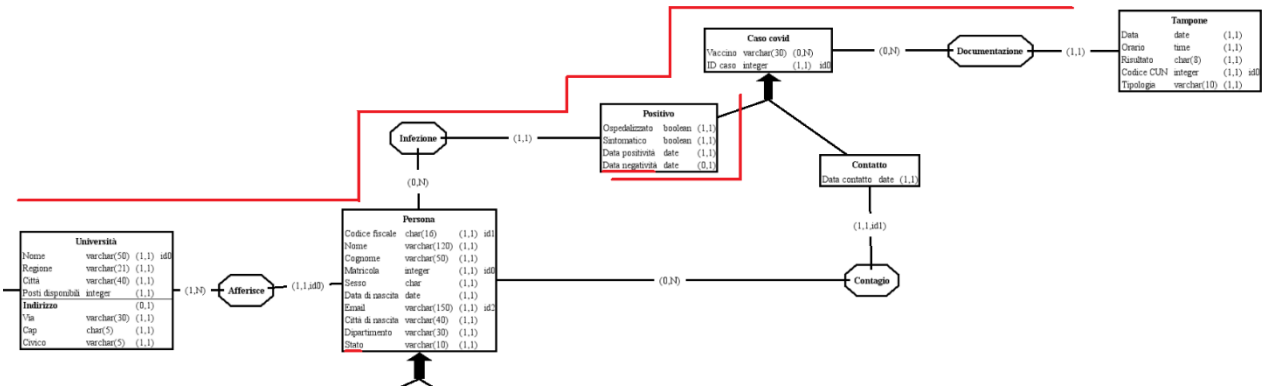
CONCETTO	COSTRUTTO	ACCESSI	TIPO
UNIVERSITA’	E	1	L
AFFERISCE	R	1	L
PERSONA	E	1	L
INFEZIONE	R	1	L
POSITIVO	E	1	L
CASO COVID	E	1	L
DOCUMENTAZIONE	R	1	L
TAMPONE	E	1	L
UNIVERSITA’	E	1	L
AFFERISCE	R	1	L
PERSONA	E	1	L
CONTAGIO	R	10	L
CONTATTO	E	10	L
CASO COVID	E	10	L
DOCUMENTAZIONE	R	10	L
TAMPONE	E	10	L



- **Operazione 2:** Per università, aggiornamento caso covid negativo

Con Ridondanza

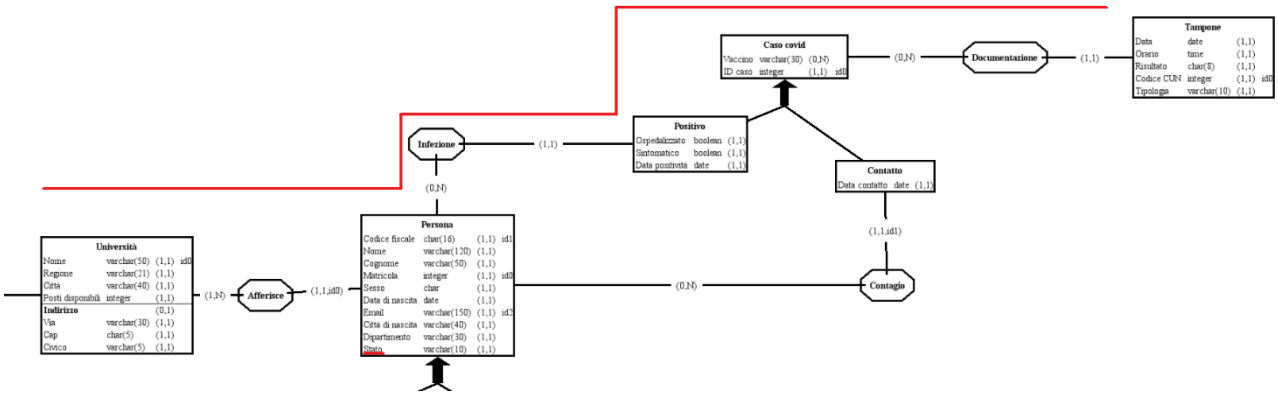
CONCETTO	COSTRUTTO	ACCESSI	TIPO
UNIVERSITA’	E	1	L
AFFERISCE	R	15200	L
PERSONA	E	15200	L
INFEZIONE	R	270	L
POSITIVO	E	270	L
CASO COVID	E	270	L
DOCUMENTAZIONE	R	15	L
TAMPONE	E	15	L
CASO COVID	E	15	L
POSITIVO	E	15	L
POSITIVO	E	15	S
PERSONA	E	15	L
PERSONA	E	15	S



Nota* = Abbiamo considerato, basandoci sulla tavola dei volumi, che in un periodo di 21 giorni in un’università sono positive circa 270 persone, di cui 15 nuove ogni giorno.

Senza Ridondanza

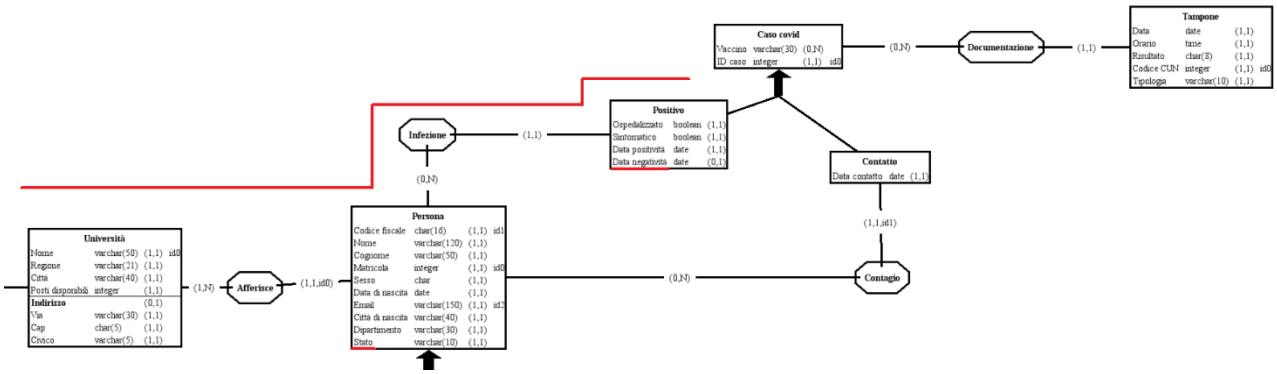
CONCETTO	COSTRUTTO	ACCESSI	TIPO
UNIVERSITA’	E	1	L
AFFERISCE	R	15200	L
PERSONA	E	15200	L
INFEZIONE	E	270	L
POSITIVO	R	270	L
CASO COVID	E	270	L
DOCUMENTAZIONE	E	15	L
TAMPONE	R	15	L
PERSONA	E	15	L
PERSONA	E	15	S



- **Operazione 7:** Per università, ricerca positivi da più di 21 giorni

Con Ridondanza

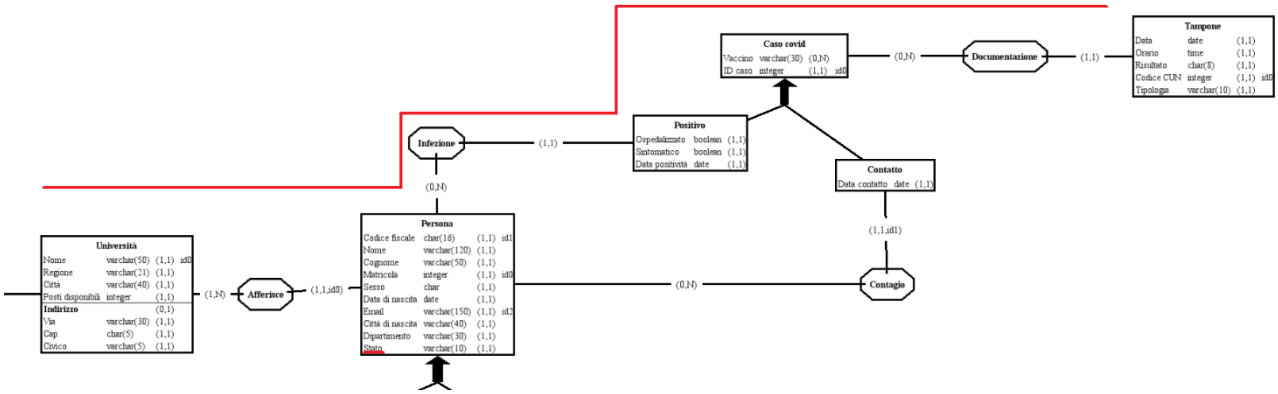
CONCETTO	COSTRUTTO	ACCESSI	TIPO
UNIVERSITA’	E	1	L
AFFERISCE	R	15200	L
PERSONA	E	15200	L
INFEZIONE	R	15	L
POSITIVO	E	15	L
CASO COVID	E	15	L
POSITIVO	R	15	L
POSITIVO	E	15	S
PERSONA	E	15	L
PERSONA	E	15	S



Nota* = Abbiamo considerato, basandoci sulla tavola dei volumi, che in un periodo di 21 giorni in un'università sono positive circa 270 persone, di cui 15 nuove ogni giorno, di conseguenza dato che questa è un'operazione giornaliera, i positivi da più di 21 giorni saranno soltanto quelli del 22esimo giorno, dato che gli altri saranno già stati resi negativi dall'operazione del giorno precedente; quindi, anche in questo caso abbiamo considerato 15 positivi.

Senza Ridondanza

CONCETTO	COSTRUTTO	ACCESSI	TIPO
UNIVERSITA'	E	1	L
AFFERISCE	R	15200	L
PERSONA	E	15200	L
INFEZIONE	E	15	L
POSITIVO	R	15	L
CASO COVID	E	15	L
DOCUMENTAZIONE	R	15	L
TAMPONE	E	15	L
PERSONA	E	15	L
PERSONA	E	15	S



4.2.1.1. Valutazione della ridondanza 1

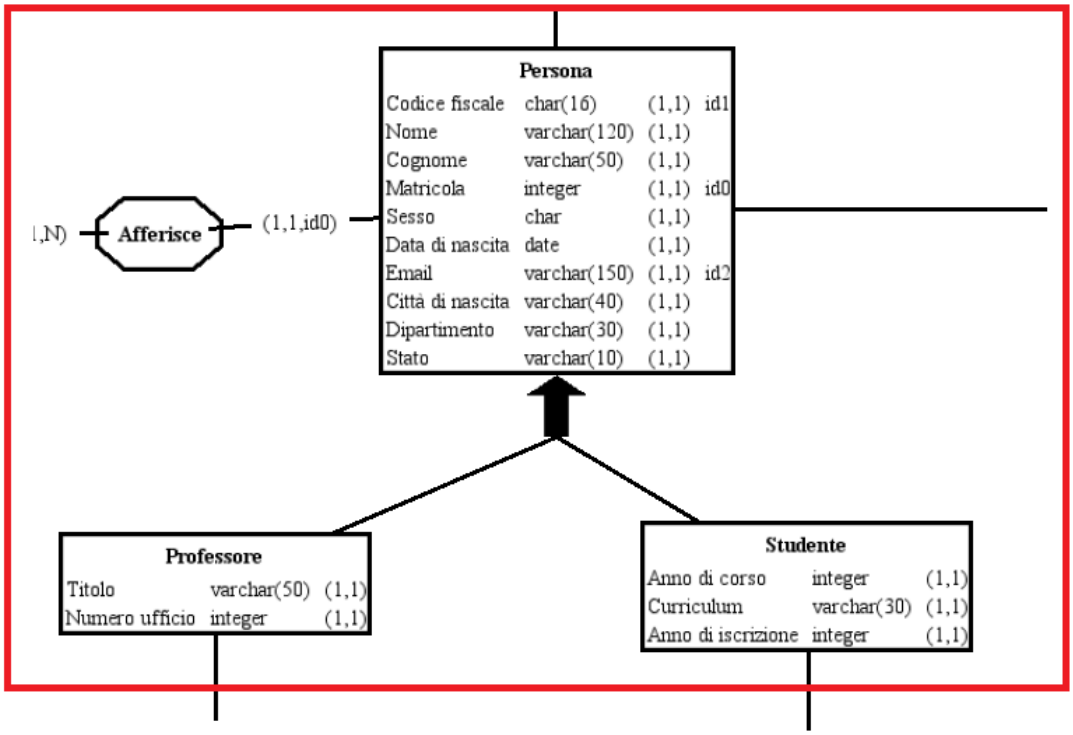
Dopo aver analizzato le operazioni che coinvolgono la ridondanza si osserva che, con il carico considerato:

- In presenza di ridondanza il costo delle operazioni è di circa 6 418 471 accessi giornalieri
- L'occupazione di memoria è di circa 1350000 byte
- In assenza di ridondanza il costo delle operazioni è di 6 415 985 accessi giornalieri

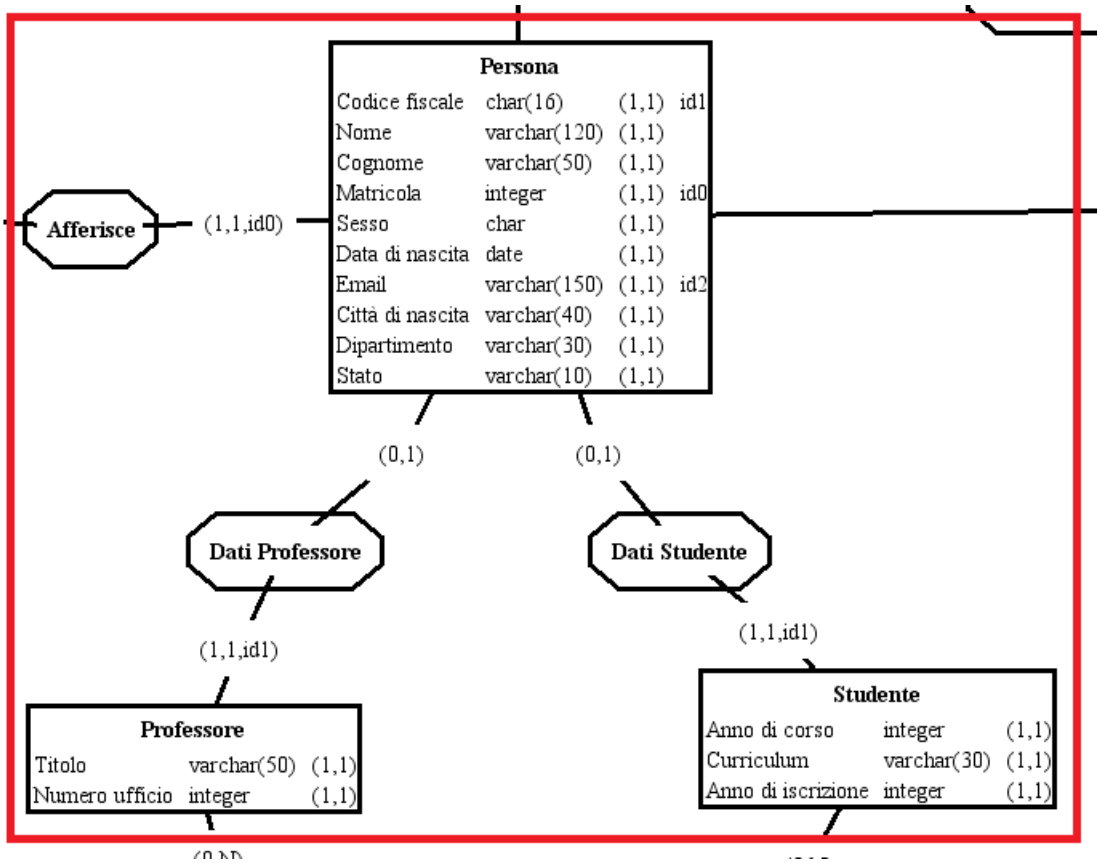
Pertanto, si decide di eliminare la ridondanza in quanto non solo aumenta, seppur in modo lieve, il numero di accessi, ma occupa anche 1350000 byte di memoria.

4.3. Eliminazione delle generalizzazioni

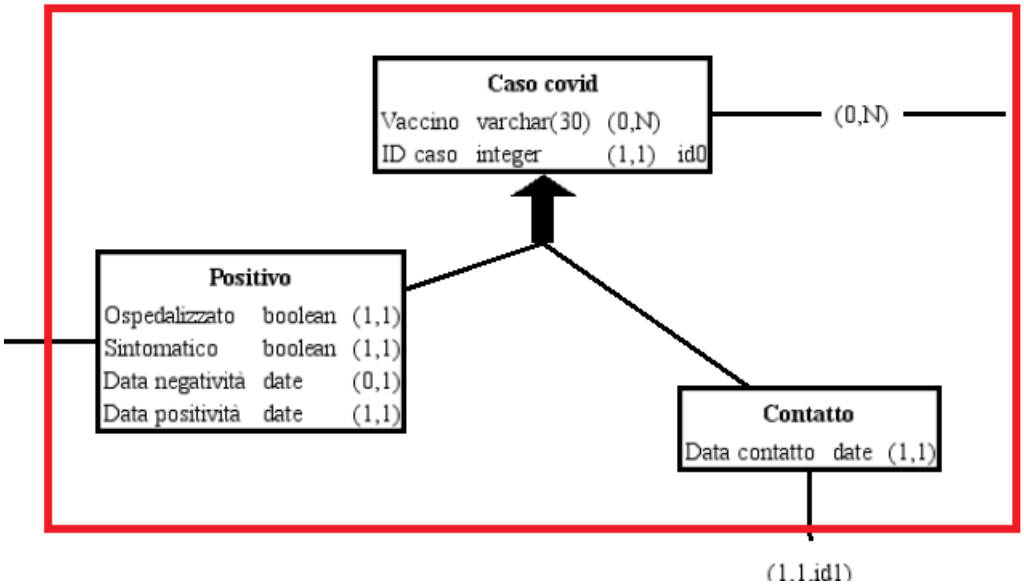
4.3.1. Generalizzazione *Persona*



La generalizzazione di persona viene eliminata usando la strategia di sostituzione della generalizzazione con associazioni. La motivazione per la quale abbiamo scelto questa strategia è perché alcune operazioni non fanno distinzione tra le due entità figlie, altre invece dipendono fortemente da questa distinzione. Inoltre, la presenza di un discreto numero di attributi nelle due entità rende preferibile lasciare le due entità separate così da evitare attributi con possibili valori nulli nell'entità padre e riducendo la dimensione delle relazioni e lo spreco di memoria.

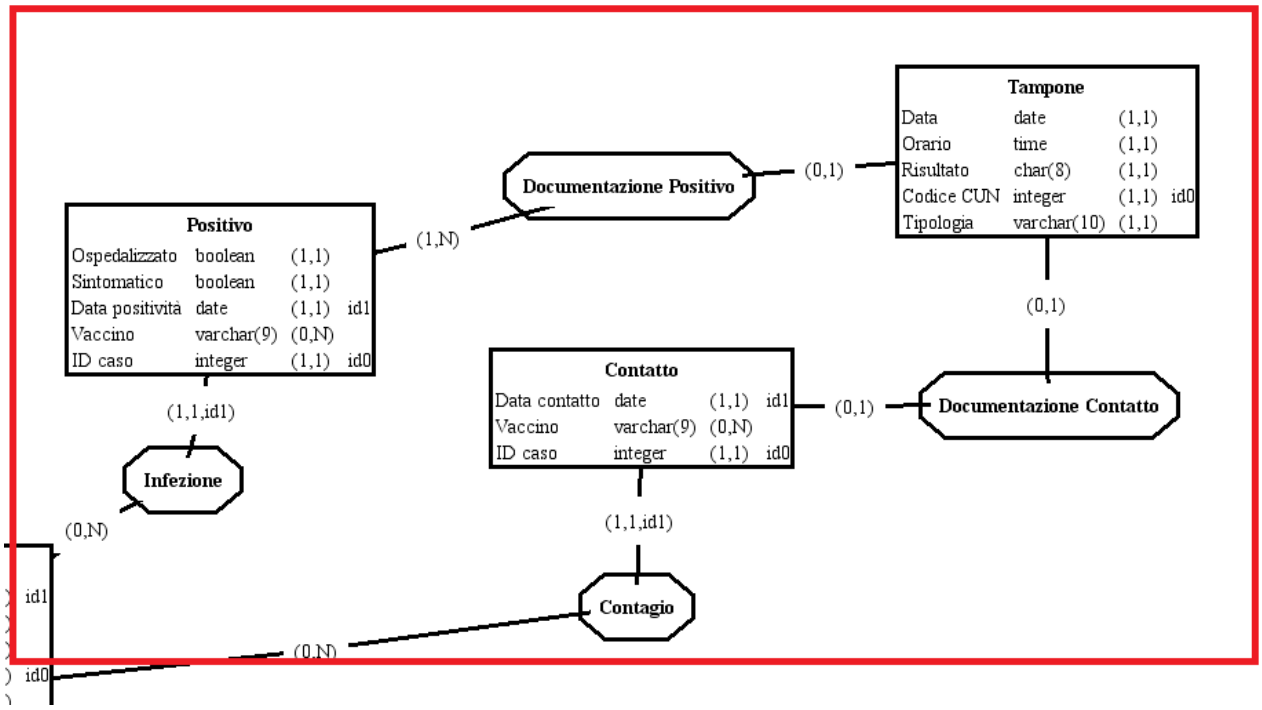


4.3.2. Generalizzazione *Caso Covid*



La generalizzazione di persona viene eliminata usando la strategia di Accorpamento del genitore della generalizzazione nelle figlie. La motivazione per la quale abbiamo scelto questa strategia è perché essa è conveniente quando le operazioni si riferiscono solo ad occorrenze di una figlia o dell'altra: Positivo e Contatto, nel nostro caso infatti, le operazioni riguardano o un'entità o un'altra. C'è inoltre un risparmio di memoria rispetto alle altre strategie poiché non ci saranno attributi nulli

relativi all'accorpamento. Inoltre, in seguito all'eliminazione, ci siamo resi conto di un'ulteriore identificativo per l'entità Positivo ovvero Data positività e esternamente attraverso l'identificativo della Persona infetta, e anche per Contatto vale la stessa cosa con Data contatto.



4.4. Partizionamento/Accorpamento Entità e Associazioni

Abbiamo deciso di non effettuare partizionamenti/accorpamenti poiché non ritenuti necessari. Ad esempio, la possibilità di partizionare l'entità Persona in dati anagrafici e dati universitari è stata ritenuta superflua poiché le operazioni che coinvolgono Persona non richiedono in maniera differita i dati, bensì sfruttano i vari attributi. Inoltre, abbiamo eliminato l'attributo multivalore Vaccino poiché un caso covid potrebbe aver eseguito fino a 3 vaccini diversi, inoltre dato che lo stesso caso covid potrebbe aver effettuato lo stesso vaccino più volte, abbiamo reificato la relazione che si viene a creare (Attestato Positivo o Attestato Contatto) rendendole entità.

4.5. Scelta degli identificatori principali

Le entità attività, calendario e aula presentano identificativi esterni che risultano essere pesanti per il sistema. Abbiamo quindi deciso di inserire in aula un codice identificativo della stessa che rimpiazzi l'identificazione esterna grazie ad Università. Riteniamo questa la scelta migliore, dato anche il numero di occorrenze che stiamo trattando. Un ragionamento simile è stato utilizzato anche per l'entità Calendario, un'identificazione tramite due attributi: Orario inizio e giorno della settimana e in più grazie all'identificatore esterno dell'aula. Ne risulta un identificatore piuttosto pesante e abbiamo deciso anche in questo caso di aggiungere un codice di riferimento per l'attività di calendario. Avendo eliminato questo "ciclo" di identificativi esterni, abbiamo, infine, constatato che per l'entità Attività un identificatore del tipo Data svolgimento insieme all'identificatore esterno di Calendario è accettabile e preferibile all'introduzione di un ulteriore codice. Un'ultima considerazione da fare riguarda l'entità Persona, Positivo e Contatto. La prima presenta tre identificatori: codice fiscale, e-mail e matricola, la cui ultima identifica la persona grazie all'Università a cui appartiene. Tra i tre abbiamo deciso di optare per il codice fiscale dato che,

nonostante sia alfanumerico e non numerico come la matricola, è comunque un identificatore interno ed è da preferire a quello esterno, inoltre lo preferiamo all'e-mail data la lunghezza minore rispetto a quest'ultima. Un ragionamento simile può essere applicato alle altre due entità, infatti abbiamo un identificatore interno, che per entrambi è rappresentato da ID caso, ed un identificatore esterno composto dalla Persona e dalla data di contatto o data di positività. Scegliamo anche in questo caso l'identificatore interno. Per le tabelle Attestato contatto e Attestato positivo dato che derivano da una reificazione di una relazione hanno come identificatori la data di somministrazione e gli identificativi delle due entità da cui deriva la relazione reificata. Per tutte le altre entità non nominate, abbiamo scelto l'unico identificativo di cui esse dispongono: per Università il Nome, per Studente e Professore l'identificazione esterna con Persona e per il Tampone il codice CUN.

4.6. Schema ristrutturato finale

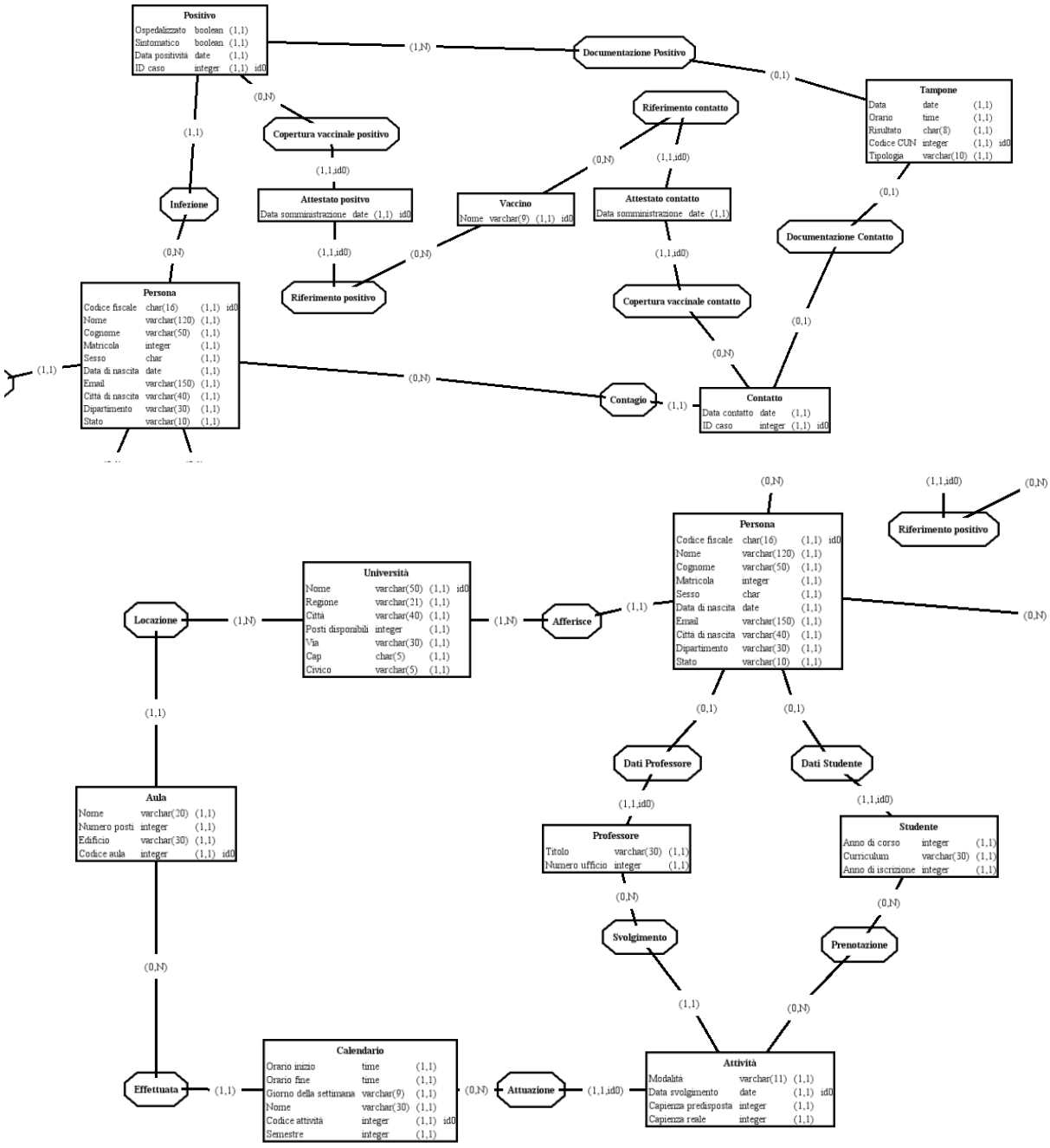


Figura 4. Schema ER Ristrutturato

4.7. Schema logico

PERSONA (Codice fiscale, Nome, Cognome, Matricola, Sesso, Data di nascita, Città di nascita, E-mail, Dipartimento, Stato, Università)

UNIVERSITA' (Nome, Regione, Città, Posti disponibili, Via, Cap, Civico)

AULA (Codice aula, Nome, Numero Posti, Edificio, Università)

CALENDARIO (Codice attività, Orario inizio, Orario fine, Giorno della settimana, Nome, Semestre, Aula)

ATTIVITA' (Data svolgimento, Attività Calendario, Modalità, Capienza predisposta, Capienza reale, Professore)

PROFESSORE (Persona, Titolo, Numero ufficio)

PRENOTAZIONE (Studente, Data attività, Attività)

STUDENTE (Persona, Anno di corso, Curriculum, Anno di iscrizione)

POSITIVO (ID caso, Ospedalizzato, Sintomatico, Data positività, Persona)

ATTESTATO POSITIVO (ID caso, Vaccino, Data somministrazione)

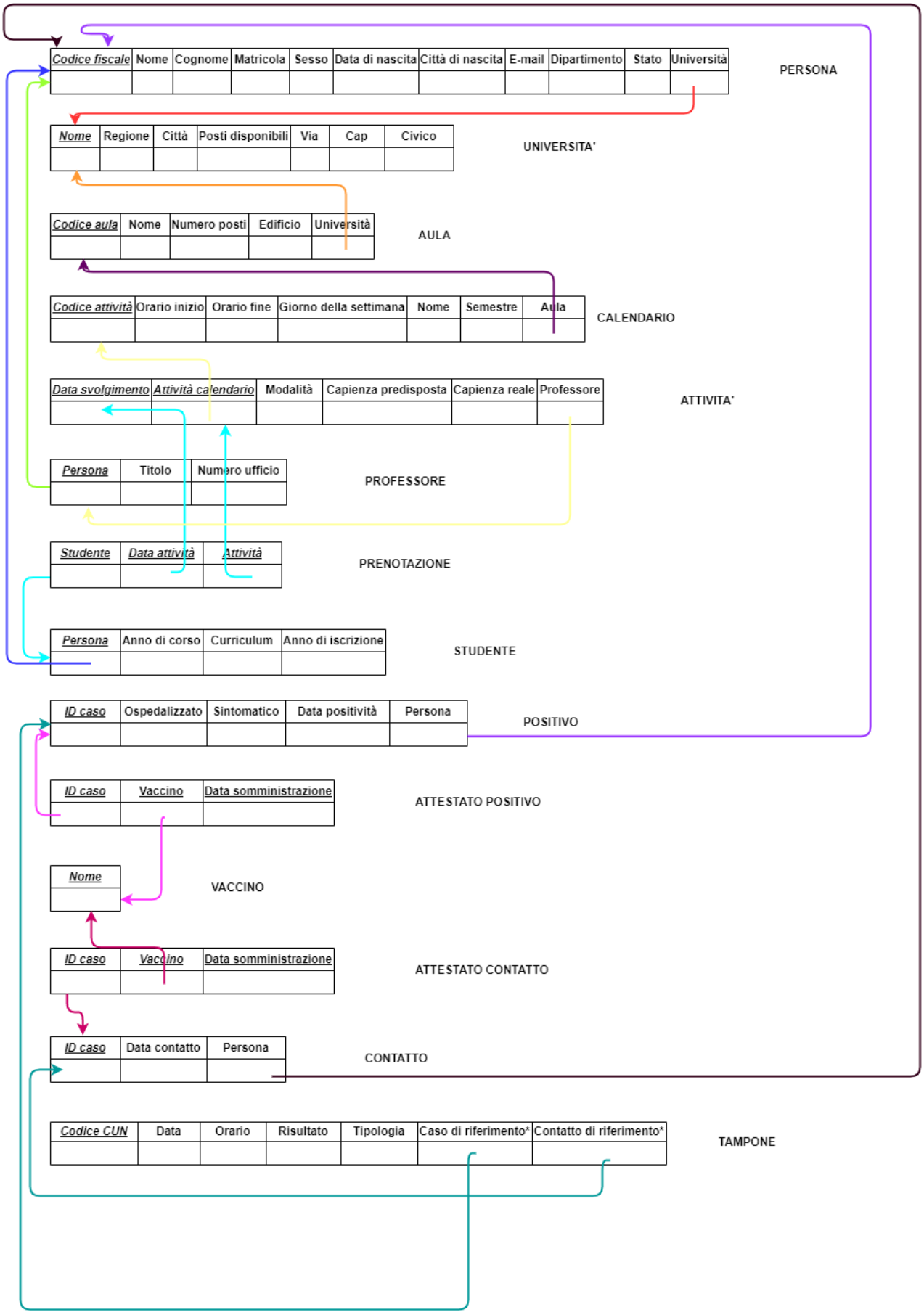
VACCINO (Nome)

ATTESTATO CONTATTO (ID caso, Vaccino, Data somministrazione)

CONTATTO (ID caso, Data contatto, Persona)

TAMPONE (Codice CUN, Data, Orario, Risultato, Tipologia, Caso di riferimento*, Contatto di riferimento*)

4.8. Documentazione dello schema logico


Codice fiscale | Nome | Cognome | Matricola | Sesso | Data di nascita | Città di nascita | E-mail | Dipartimento | Stato | Università
PERSONA
Nome | Regione | Città | Posti disponibili | Via | Cap | Civico
UNIVERSITA'
Codice aula | Nome | Numero posti | Edificio | Università
AULA
Codice attività | Orario inizio | Orario fine | Giorno della settimana | Nome | Semestre | Aula
CALENDARIO
Data svolgimento | Attività calendario | Modalità | Capienza predisposta | Capienza reale | Professore
ATTIVITA'
Persona | Titolo | Numero ufficio
PROFESSORE
Studente | Data attività | Attività
PRENOTAZIONE
Persona | Anno di corso | Curriculum | Anno di iscrizione
STUDENTE
ID caso | Ospedalizzato | Sintomatico | Data positività | Persona
POSITIVO
ID caso | Vaccino | Data somministrazione
ATTESTATO POSITIVO
Nome
VACCINO
ID caso | Vaccino | Data somministrazione
ATTESTATO CONTATTO
ID caso | Data contatto | Persona
CONTATTO
Codice CUN | Data | Orario | Risultato | Tipologia | Caso di riferimento* | Contatto di riferimento*
TAMPONE

Nota: Dato il passaggio dallo schema concettuale allo schema logico dobbiamo inserire ulteriori vincoli:

- Data la scomposizione della generalizzazione di Persona: L'entità persona deve partecipare o ad un'occorrenza di Professore o ad un'occorrenza di Studente e solo ad una delle due.
- Un Positivo deve avere il riferimento al tampone positivo per esistere.
- Il tampone deve avere un caso di riferimento o un contatto di riferimento e solo uno dei due.
- L'università deve avere almeno un professore, un'aula e uno studente.

5. Normalizzazione

Workpackage	***Task***	***Responsabile***
WP3	Normalizzazione	Tarantino Ramona

*I requisiti per la prima forma normale (1FN) richiedono che il dominio degli attributi comprenda soltanto valori indivisibili e inoltre che il valore degli attributi sia un valore singolo del dominio di quel rispettivo attributo. Nel nostro caso, per ogni tabella, raggiungiamo la 1FN. Il rispetto di questa forma deriva anche dal fatto che la prima forma normale ha requisiti concordi con quella che è la definizione di relazione nel modello relazionale, infatti molti requisiti, tra cui quello per cui le righe della tabella contengano lo stesso numero di colonne e i cui valori all'interno appartengano allo stesso dominio, attributi che contengano informazioni elementari e che ogni riga sia diversa da tutte le altre senza basarsi su nessun ordinamento, rispecchiano perfettamente quelli richiesti dalla 1FN. Il raggiungimento della seconda forma normale (2FN) richiede, implicitamente, che le tabelle siano già in 1FN e anche che gli attributi non facenti parte di una chiave della tabella devono dipendere (nel senso di dipendenza funzionale) dalla chiave nella sua interezza. In pratica non devono esistere attributi che dipendono soltanto da una parte della chiave. Questa forma normale è quindi verificata a priori in quelle tabelle dove già è verificata la 1FN e la chiave è costituita da un solo attributo. Questo è il caso della maggior parte delle nostre tabelle, tranne per la tabella Attività nella quale la chiave è costituita da due attributi: Data svolgimento e Attività Calendario. Dobbiamo quindi verificare che i restanti attributi dipendano dall'intera chiave e non soltanto da uno dei due attributi. Dall'analisi effettuata risulta che tutti e tre gli attributi non chiave: Modalità, Capienza Predisposta e Capienza Reale non dipendono unicamente da uno dei due attributi, o meglio, si potrebbe pensare che esista la dipendenza Attività Calendario -> Capienza reale, ma abbiamo considerato anche il caso in cui un'aula potrebbe variare nel tempo la sua capienza (in seguito a lavori di manutenzione), di conseguenza la sola attività di calendario non è abbastanza per derivare univocamente la capienza reale, ma ha bisogno anche della data di svolgimento, di conseguenza anche Attività è in 2FN. Per quanto riguarda invece le tabelle derivanti da associazioni, essendo binarie, sono, secondo lo studio teorico, già in terza forma normale (3FN), ma per essere precisi, la tabella Prenotazione ha attributi che sono tutti chiave, quindi è sicuramente in 2FN, per quanto riguarda le tabelle Attestato * esse nascono dalla reificazione della relazione e hanno come chiave l'insieme di tutti gli attributi, di conseguenza sono in 2FN. Passiamo ora alla 3FN, essa richiede, implicitamente, la verifica della 2FN, e ulteriormente richiede che gli attributi non facenti parte di una chiave della tabella debbano dipendere direttamente dalla chiave e non da altri attributi che non lo siano, ovvero non devono dipendere transitivamente dalla chiave. Analizziamo tabella per tabella. La tabella Persona ha tre chiavi: codice fiscale, che è anche chiave primaria, la coppia di matricola e università e infine l'e-mail. Gli attributi non chiave sono: Nome, Cognome, Sesso, Data di nascita, Città di nascita,*

*Dipartimento e Stato. Dall'analisi effettuata nessuno di questi attributi è determinante per altri, ciò significa che tutti gli attributi dipendono da attributi che sono chiave, siamo in 3FN. La tabella Università ha una chiave primaria, Nome, e un'altra chiave rappresentata dall'insieme degli attributi Via, Cap e Civico. Gli attributi non chiave: Regione, Città e Posti Disponibili non dipendono da attributi che non siano chiave per Università, di conseguenza anche in questo caso abbiamo raggiunto la 3FN. Passiamo ora ad Aula, la chiave primaria in questo caso è Codice aula, mentre un'altra chiave è l'insieme di Nome, Edificio e Università. Dato che l'unico attributo non chiave è Numero posti, esso sicuramente dipende da una chiave, 3FN anche in questo caso. Calendario, invece, ha come chiave primaria il codice dell'attività ed un ulteriore chiave rappresentata da Orario inizio, Giorno della settimana e Aula. Gli attributi non chiave sono: Orario fine, Nome e Semestre. Non essendoci dipendenze funzionali in cui il determinante non sia il codice, siamo in 3FN anche per la tabella Calendario. Ora analizziamo la tabella Attività, essa ha come chiave primaria, e anche unica chiave, la coppia Data svolgimento e Attività Calendario. Il restante degli attributi, ovvero: Modalità, Capienza predisposta, Capienza reale e Professore dipendono dalla chiave stessa e solamente da essa, di conseguenza 3FN raggiunta anche qui. La tabella Professore è molto semplice, la chiave primaria è Persona, e gli altri due attributi Titolo e Numero ufficio non hanno dipendenze fra di loro, anche in questo caso siamo in 3FN. Come Professore anche Studente è una tabella semplice, nel senso che possiede pochi attributi, come chiave primaria abbiamo anche in questo caso Persona, e i restanti tre attributi Anno di corso, Curriculum e Anno di iscrizione non dipendono da attributi che non siano Persona. Facciamo notare che non esiste una dipendenza tra Anno di iscrizione e Anno di corso poiché non è detto che una persona iscritta, per esempio, 3 anni fa ora stia seguendo il terzo anno. La tabella Positivo ha come chiave primaria ID caso e ha come ulteriore chiave la coppia Persona e Data positività. Il restante degli attributi comprende Ospedalizzato e Sintomatico; nessuno di questi attributi dipende dall'altro, ovvero dipendono tutti da una chiave, abbiamo raggiunto la 3FN. Lo stesso ragionamento è condivisibile anche per Contatto, in cui la chiave primaria è ID caso e presenta un ulteriore chiave rappresentata da Data Contatto e Persona, non essendoci altri attributi, la 3FN è rispettata. Abbiamo anche la tabella Vaccino, che, avendo un unico attributo che necessariamente è chiave primaria, è in 3FN. Infine, abbiamo la tabella Tampone le cui occorrenze sono identificate univocamente da Codice CUN. Altri attributi tra cui Data, Orario, Risultato, Tipologia, Caso di riferimento, Contatto di riferimento dipendono soltanto dalla chiave, abbiamo raggiunto la 3FN ovunque. Anche in questo caso non è fondamentale analizzare le tabelle derivanti da associazioni perché esse si troveranno sicuramente già in 3FN, considerando anche Attestato * derivante dalla reificazione che soltanto attributi chiave ed è in 3FN anch'essa. Infine, verifichiamo se le nostre tabelle soddisfano la forma normale di Boyce e Codd (BCNF), essa è una forma un po' più forte della 3FN, infatti essere in BCNF implica essere in 3FN, e la differenza con la sua predecessora sta nel fatto che la BCNF richiede che ogni attributo dell'insieme non deve dipendere da attributi non chiave. Dalle analisi svolte precedentemente ci si rende conto che in realtà la forma normale raggiunta dal nostro schema è effettivamente la BCNF.*

6. Script Creazione e Popolamento Database

Workpackage	***Task***	***Responsabile***
WP2	SQL: Script creazione e popolamento	Venditti Francesca

CREAZIONE DELLE TABELLE

```
DROP TABLE IF EXISTS Università CASCADE;
CREATE TABLE Università(
    Nome varchar(50),
    Regione varchar(21) NOT NULL,
    Città varchar(40) NOT NULL,
    Posti_disponibili integer NOT NULL CONSTRAINT posti_positivi CHECK(Posti_disponibili > 0),
    Via varchar(30) NOT NULL,
    Cap char(5) NOT NULL,
    Civico varchar(5) NOT NULL,
    CONSTRAINT pk_università PRIMARY KEY(Nome),
    UNIQUE(Via,Cap,Civico)
);

DROP TABLE IF EXISTS Aula CASCADE;
CREATE TABLE Aula(
    Codice_aula serial,
    Nome varchar(20) NOT NULL,
    Numero_posti integer NOT NULL,
    Edificio varchar(30) NOT NULL,
    Università varchar(50) NOT NULL,
    CONSTRAINT pk_aula PRIMARY KEY(Codice_aula),
    CONSTRAINT fk_aula_università FOREIGN KEY(Università) references Università(Nome) on update cascade on delete restrict deferrable initially deferred,
    UNIQUE(Nome,Edificio,Università)
);

DROP TABLE IF EXISTS Persona CASCADE;
CREATE TABLE Persona(
    Codice_fiscale char(16),
    Nome varchar(120) NOT NULL,
    Cognome varchar(50) NOT NULL,
    Matricola integer NOT NULL,
    Sesso char NOT NULL,
    Data_di_nascita date NOT NULL,
    Città_di_nascita varchar(40) NOT NULL,
    Email varchar(150) NOT NULL UNIQUE,
    Dipartimento varchar(30) NOT NULL,
    Stato varchar(10) NOT NULL DEFAULT 'Libero',
    Università varchar(50) NOT NULL,
    CONSTRAINT fk_persona_università FOREIGN KEY(Università) references Università(Nome) on update cascade on delete restrict deferrable initially deferred,
    CONSTRAINT pk_persona PRIMARY KEY(Codice_fiscale),
    UNIQUE(Matricola,Università),
    CONSTRAINT check_email CHECK (Email LIKE '%_@__%.__%'),
    CONSTRAINT check_data_nascita CHECK (Data_di_nascita > '1930-01-01' and Data_di_nascita <  CURRENT_DATE-17*365)
);
```

```
DROP DOMAIN IF EXISTS Settimana cascade;
CREATE DOMAIN Settimana as varchar(9)
CHECK (VALUE = 'Lunedì' or VALUE = 'Martedì' or VALUE = 'Mercoledì' or VALUE = 'Giovedì' or VALUE = 'Venerdì')
NOT NULL;

DROP TABLE IF EXISTS Calendario CASCADE;
CREATE TABLE Calendario(
    Codice_attività serial,
    Orario_inizio time NOT NULL,
    Orario_fine time NOT NULL,
    Giorno_settimana Settimana,
    Nome varchar(30) NOT NULL,
    Semestre integer NOT NULL,
    Aula integer NOT NULL,
    CONSTRAINT pk_calendario PRIMARY KEY(Codice_attività),
    CONSTRAINT check_giorno_settimana CHECK (Giorno_settimana != 'Sabato' and Giorno_settimana !='Domenica'),
    CONSTRAINT check_durata CHECK (Orario_fine > Orario_inizio),
    CONSTRAINT check_semestre CHECK(Semestre = 1 or Semestre = 2),
    CONSTRAINT fk_calendario_aula FOREIGN KEY(Aula) references Aula(Codice_aula) on update cascade on delete set null
);

DROP TABLE IF EXISTS Professore CASCADE;
CREATE TABLE Professore(
    Persona char(16),
    Titolo varchar(50) NOT NULL,
    Numero_ufficio integer NOT NULL,
    CONSTRAINT pk_professore PRIMARY KEY(Persona),
    CONSTRAINT fk_professore_persona FOREIGN KEY(Persona) references Persona(Codice_fiscale) on update cascade on delete restrict deferrable initially deferred
);

DROP TABLE IF EXISTS Studente CASCADE;
CREATE TABLE Studente(
    Persona char(16),
    Anno_corso integer NOT NULL,
    Curriculum varchar(30) NOT NULL,
    Anno_iscrizione integer NOT NULL,
    CONSTRAINT pk_studente PRIMARY KEY(Persona),
    CONSTRAINT fk_studente_persona FOREIGN KEY(Persona) references Persona(Codice_fiscale) on update cascade on delete restrict deferrable initially deferred

);


DROP TABLE IF EXISTS Attività CASCADE;
CREATE TABLE Attività(
    Data_svolgimento date,
    Attività_calendario integer,
    Modalità varchar(11) NOT NULL DEFAULT 'Presenza',
    Capienza_predisposta integer NOT NULL,
    Capienza_reale integer NOT NULL,
    Professore char(16) NOT NULL,
    CONSTRAINT pk_attività PRIMARY KEY(Data_svolgimento,Attività_calendario),
    CONSTRAINT fk_attività_calendario FOREIGN KEY(Attività_calendario) references Calendario(Codice_attività) on update cascade on delete restrict,
    CONSTRAINT check_capienza CHECK(Capienza_predisposta > 0 AND Capienza_reale > 0 AND Capienza_reale >= Capienza_predisposta),
    CONSTRAINT check_attività_periodo CHECK(Data_svolgimento >= CURRENT_DATE),
    CONSTRAINT fk_attività_professore FOREIGN KEY(Professore) references Professore(Persona) on delete restrict on update cascade
);

DROP TABLE IF EXISTS Prenotazione CASCADE;
CREATE TABLE Prenotazione(
    Studente char(16),
    Data_attività date,
    Attività integer,
    CONSTRAINT pk_prenotazione PRIMARY KEY(Studente,Data_attività,Attività),
    CONSTRAINT fk_prenotazione_persona FOREIGN KEY(Studente) references Studente(Persona) on update cascade on delete cascade,
    CONSTRAINT fk_prenotazione_attività FOREIGN KEY(Data_attività,Attività) references Attività(Data_svolgimento,Attività_calendario) on update cascade on delete cascade
);

DROP TABLE IF EXISTS Positivo CASCADE;
CREATE TABLE Positivo(
    ID_caso serial,
    Ospedalizzato boolean NOT NULL DEFAULT False,
    Sintomatico boolean NOT NULL DEFAULT False,
    Data_positività date NOT NULL,
    Persona char(16) NOT NULL,
    CONSTRAINT pk_positivo PRIMARY KEY(ID_caso),
    CONSTRAINT fk_positivo_persona FOREIGN KEY(Persona) references Persona(Codice_fiscale) on update cascade on delete cascade,
    UNIQUE(Data_positività,Persona),
    CONSTRAINT check_data_positività CHECK (Data_positività < (CURRENT_DATE+1))
);

DROP TABLE IF EXISTS Vaccino CASCADE;
CREATE TABLE Vaccino(
    Nome varchar(9),
    CONSTRAINT pk_vaccino PRIMARY KEY(Nome)
);
```

```
DROP TABLE IF EXISTS Attestato_positivo CASCADE;
CREATE TABLE Attestato_positivo(
    ID_caso integer,
    Vaccino varchar(9),
    Data_somministrazione date NOT NULL,
    CONSTRAINT pk_attestato_positivo PRIMARY KEY(ID_caso,Vaccino,Data_somministrazione),
    CONSTRAINT fk_attestato_positivo_positivo FOREIGN KEY(ID_caso) references Positivo(ID_caso) on update cascade on delete cascade,
    CONSTRAINT fk_attestato_positivo_vaccino FOREIGN KEY(Vaccino) references Vaccino(Nome) on update restrict on delete restrict,
    CONSTRAINT check_data_somministrazione_positivo CHECK (Data_somministrazione < (CURRENT_DATE+1))
);

DROP TABLE IF EXISTS Contatto CASCADE;
CREATE TABLE Contatto(
    ID_caso serial,
    Data_contatto date NOT NULL default CURRENT_DATE,
    Persona char(16) NOT NULL,
    CONSTRAINT pk_contatto PRIMARY KEY(ID_caso),
    CONSTRAINT fk_contatto_persona FOREIGN KEY(Persona) references Persona(Codice_fiscale) on update cascade on delete cascade,
    UNIQUE(Data_contatto,Persona),
    CONSTRAINT check_data_contatto CHECK (Data_contatto < (CURRENT_DATE+1))
);
DROP TABLE IF EXISTS Attestato_contatto CASCADE;
CREATE TABLE Attestato_contatto(
    ID_caso integer,
    Vaccino varchar(9),
    Data_somministrazione date,
    CONSTRAINT pk_attestato_contatto PRIMARY KEY(ID_caso,Vaccino,Data_somministrazione),
    CONSTRAINT fk_attestato_contatto_contatto FOREIGN KEY(ID_caso) references Contatto(ID_caso) on update cascade on delete cascade,
    CONSTRAINT fk_attestato_contatto_vaccino FOREIGN KEY(Vaccino) references Vaccino(Nome) on update restrict on delete restrict,
    CONSTRAINT check_data_somministrazione_contatto CHECK (Data_somministrazione < (CURRENT_DATE+1))
);


DROP TABLE IF EXISTS Tampone CASCADE;
CREATE TABLE Tampone(
    Codice_CUN serial,
    Data date NOT NULL,
    Orario time NOT NULL,
    Risultato char(8) NOT NULL,
    Tipologia varchar(10) NOT NULL,
    Caso_riferimento integer,
    Contatto_riferimento integer,
    CONSTRAINT pk_tampone PRIMARY KEY(Codice_CUN),
    CONSTRAINT fk_tampone_positivo FOREIGN KEY(Caso_riferimento) references Positivo(ID_caso) on update cascade on delete cascade deferrable initially deferred,
    CONSTRAINT fk_tampone_contatto FOREIGN KEY(Contatto_riferimento) references Contatto(ID_caso) on update cascade on delete cascade,
    CONSTRAINT check_data_tampone CHECK (Data < (CURRENT_DATE+1))
);
```

--POPOLAMENTO

```
/*NOTA: Nonostante alcuni attributi siano seriali, li abbiamo inseriti manualmente
e non attraverso la modalità DEFAULT per facilitarci l'inserimento di altre tabelle
che
referenziavano questi attributi. Inoltre, non è presente il popolamento delle
tabelle Contatto e Attestato Contatto, dato che i primi vengono inseriti attraverso
la query di
tracciamento dei contatti e gli attestati vengono caricati in seguito alla
segnalazione del contatto.
Nel file di popolamento ci sono alcuni script a seconda del giorno della settimana
in cui viene visto il progetto. Questo perché nel nostro database il giorno delle
attività,
deve coincidere con il giorno della settimana della schedulazione nel calendario, e
dato che non possiamo inserire attività passate, abbiamo pensato di creare 5 script
diversi
a seconda del giorno della settimana in cui si sta controllando. */
BEGIN TRANSACTION;
INSERT INTO Università values ('Università degli studi di Salerno', 'Campania',
'Salerno', 30000, 'Via Giovanni Paolo', '84084', '132');
INSERT INTO Università values ('Università degli studi di Napoli', 'Campania',
'Napoli', 10000, 'Corso Umberto', '80138', '40');
```

```
INSERT INTO Università values ('Politecnico di
Milano','Lombardia','Milano',55000,'Piazza Leonardo Da Vinci','20133','32');
INSERT INTO Università values ('Università degli studi di
Firenze','Toscana','Firenze',30000,'Piazza San Marco','50121','4');
INSERT INTO Università values ('Politecnico di
Torino','Piemonte','Torino',60000,'Corso Duca Degli Abruzzi','86100','16');

--PERSONE Università degli studi di Salerno
INSERT INTO Persona values
('TRNRMN00T52A783A','Ramona','Tarantino',1612704902,'F','2000-12-
12','Benevento','rtarantino7@unisa.it','Ingegneria
Informatica','Libero','Università degli studi di Salerno');
INSERT INTO Persona values
('MLNMRC00D04C361Z','Marco','Milone',1612783940,'M','2000-12-26','Cava de
tirreni','mmilone15@unisa.it','Ingegneria Informatica','Libero','Università degli
studi di Salerno');
INSERT INTO Persona values
('VNDFNC00D68G964C','Francesca','Venditti',1612783640,'F','2000-04-
28','Pozzuoli','fvenditti1@unisa.it','Ingegneria Informatica','Libero','Università
degli studi di Salerno');
INSERT INTO Persona values
('NTNGNN01B21A399J','Giovanni','Intonti',1612783899,'M','2001-02-21','Ariano
Irpino','gintonti@unisa.it','Ingegneria Informatica','Libero','Università degli
studi di Salerno');
INSERT INTO Persona values
('VTLLCU98B16F839G','Luca','Vitale',1612704910,'M','1998-02-
16','Napoli','lvitale9@unisa.it','Farmacia','Libero','Università degli studi di
Salerno');
INSERT INTO Persona values
('MNZLSE02L53B963F','Elisa','Manzo',1612783789,'F','2002-07-
03','Caserta','emanzo8@unisa.it','Lettere','Libero','Università degli studi di
Salerno');
INSERT INTO Persona values ('DLCVCN97L05H703I','Vincenzo','De
Luca',1612783543,'M','1997-07-
05','Salerno','vdeluca@unisa.it','Giurisprudenza','Libero','Università degli studi
di Salerno');
INSERT INTO Persona values
('MRCMTT99L17A509P','Matteo','Marcone',1612783874,'M','1999-07-
17','Avellino','mmarcone@unisa.it','Economia','Libero','Università degli studi di
Salerno');
INSERT INTO Persona values
('CSNRKE00C70F839Y','Erika','Cusano',1612783876,'F','2000-03-
30','Napoli','ecusano@unisa.it','Matematica','Libero','Università degli studi di
Salerno');
INSERT INTO Persona values ('CSCGAI00D67H703B','Gaia','Casci',1612783664,'F','2000-
04-27','Salerno','gcasci@unisa.it','Economia','Libero','Università degli studi di
Salerno');
```

```
INSERT INTO Persona values
('GTAMTT60L17H703S','Matteo','Gaeta',1000000000,'M','1960-05-
16','Salerno','mgaeta@unisa.it','Ingegneria Informatica','Libero','Università degli
studi di Salerno');
INSERT INTO Persona values
('GRCGPP75D08H703K','Giuseppe','Greco',1000000001,'M','1975-04-
08','Salerno','ggreco@unisa.it','Fisica','Libero','Università degli studi di
Salerno');
INSERT INTO Persona values
('MNZRNN70A48F839L','Rosanna','Manzo',1000000002,'F','1972-04-
27','Napoli','rmanzo@unisa.it','Economia','Libero','Università degli studi di
Salerno');

--PERSONE Università degli studi di Napoli
INSERT INTO Persona values
('RSSMTT00S12F839U','Matteo','Rossi',1000099999,'M','2000-12-
11','Napoli','mrossi@unina.it','Ingegneria Informatica','Libero','Università degli
studi di Napoli');
INSERT INTO Persona values
('VRDMRC00S12F839T','Marco','Verdi',1000099998,'M','2000-12-22','Torre del
Greco','mverdi@unina.it','Ingegneria Chimica','Libero','Università degli studi di
Napoli');
INSERT INTO Persona values
('NREGPP00S12B990R','Giuseppe','Neri',1000099997,'M','2000-04-
18','Casoria','gneri@unina.it','Ingegneria Informatica','Libero','Università degli
studi di Napoli');
INSERT INTO Persona values
('FRRLLN00S52A064B','Liliana','Ferrari',1000099910,'F','2001-02-
28','Afragola','lferrari8@unina.it','Ingegneria Informatica','Libero','Università
degli studi di Napoli');
INSERT INTO Persona values
('RSSFNC01S52F839H','Franca','Russo',1000099911,'F','1998-02-
11','Napoli','frusso1@unina.it','Farmacia','Libero','Università degli studi di
Napoli');
INSERT INTO Persona values
('RMNDBR98A41B963Z','Debora','Romano',1000099912,'F','2002-07-
09','Caserta','dromano998@unina.it','Farmacia','Libero','Università degli studi di
Napoli');
INSERT INTO Persona values ('DLCLNR97A41B963J','Eleonora','De
Luca',1000099913,'F','1997-07-
05','Ercolano','edeluca@unina.it','Matematica','Libero','Università degli studi di
Napoli');
INSERT INTO Persona values
('FNTMTT99A01A509U','Matteo','Fontana',1000099914,'M','1999-07-
12','Avellino','mfontana3@unina.it','Economia','Libero','Università degli studi di
Napoli');
INSERT INTO Persona values ('CSTLCU00A01H703F','Luca','Costa',1000099915,'M','2000-
03-23','Acerra','lcosta@unina.it','Matematica','Libero','Università degli studi di
Napoli');
```

```
INSERT INTO Persona values ('DFLPLA00A01H703Q','Paolo','De
Filippo',1000099916,'M','2000-04-
21','Salerno','pdefilippo6@unina.it','Economia','Libero','Università degli studi di
Napoli');

INSERT INTO Persona values
('PACRCC60L17H703S','Riccardo','Pace',1000000003,'M','1965-07-
19','Salerno','rpace@unina.it','Ingegneria Informatica','Libero','Università degli
studi di Napoli');
INSERT INTO Persona values
('MORGSP75D08H703K','Giuseppe','Moro',1000000004,'M','1975-03-
08','Benevento','gmoro@unina.it','Matematica','Libero','Università degli studi di
Napoli');
INSERT INTO Persona values
('MNCMNN70A48F839T','Marianna','Mancini',1000000005,'F','1978-10-
27','Napoli','mmancini@unina.it','Economia','Libero','Università degli studi di
Napoli');

--PERSONE Politecnico di Milano
INSERT INTO Persona values
('TFNHRD63R03B217K','Lorenzo','Colombo',1002222222,'M','2000-12-
11','Milano','lcolombo@polimi.it','Ingegneria Informatica','Libero','Politecnico di
Milano');
INSERT INTO Persona values
('MLKBZR68M42B350G','Gabriele','Sala',1002222223,'M','2001-10-
12','Milano','gsala@polimi.it','Ingegneria Chimica','Libero','Politecnico di
Milano');
INSERT INTO Persona values
('ZTQTFR56A58L697L','Tommaso','Villa',1002222224,'M','2002-09-
10','Brescia','tvilla@polimi.it','Ingegneria Industriale','Libero','Politecnico di
Milano');
INSERT INTO Persona values
('SWFRYF37B65A948O','Alessandro','Cattaneo',1002222225,'M','1999-08-
09','Monza','acattaneo@polimi.it','Ingegneria Informatica','Libero','Politecnico di
Milano');
INSERT INTO Persona values
('QVTDNZ40L07C348V','Edoardo','Brambilla',1002222226,'M','2000-07-
20','Como','ebrambilla@polimi.it','Economia','Libero','Politecnico di Milano');
INSERT INTO Persona values
('YMCBTX40A41I605Z','Sofia','Rossi',1002222227,'F','1998-06-
22','Milano','srossi22@polimi.it','Economia','Libero','Politecnico di Milano');
INSERT INTO Persona values
('VFCZLG94R66M271D','Aurora','Fumagalli',1002222228,'F','2002-05-
30','Varese','afumagalli@polimi.it','Ingegneria Chimica','Libero','Politecnico di
Milano');
INSERT INTO Persona values
('RTGMTL30P44G691C','Giulia','Riva',1002222229,'F','1996-04-
12','Pavia','griva@polimi.it','Ingegneria Chimica','Libero','Politecnico di
Milano');
```

```
INSERT INTO Persona values
('TQBMNN79D44G997S','Ginevra','Bianchi',1002222210,'F','1997-12-
24','Cremona','gbianchi76@polimi.it','Economia','Libero','Politecnico di Milano');
INSERT INTO Persona values ('MDMPTN91M02F527H','Emma','Russo',1002222211,'F','2001-
11-08','Mantova','erusso6@polimi.it','Economia','Libero','Politecnico di Milano');

INSERT INTO Persona values
('CSTPPY83B15E317O','Stefano','Palladino',1000000006,'M','1969-07-
19','Monza','spalladino@polimi.it','Ingegneria Informatica','Libero','Politecnico
di Milano');
INSERT INTO Persona values
('SNBZFC36C47H334A','Fabrizio','Ferrero',1000000007,'M','1971-07-
18','Milano','fferrero2@polimi.it','Ingegneria Chimica','Libero','Politecnico di
Milano');
INSERT INTO Persona values
('WPKRHT53L24L325A','Rossella','Caruso',1000000008,'F','1981-11-
29','Roma','rcaruso@polimi.it','Economia','Libero','Politecnico di Milano');

--PERSONE Politecnico di Torino
INSERT INTO Persona values
('ZDYFHO27M61G582Y','Samuele','Gallo',1111133333,'M','2000-11-
22','Torino','sgallo2@polito.it','Informatica','Libero','Politecnico di Torino');
INSERT INTO Persona values
('CSPDTM27E68E997W','Lorenzo','Rizzo',1111133334,'M','2001-10-
26','Asti','lrizzo2@polito.it','Biologia','Libero','Politecnico di Torino');
INSERT INTO Persona values
('MJXHPZ53C69B591C','Riccardo','Santoro',1111133335,'M','2003-09-
25','Cuneo','rsantoro1@polito.it','Informatica','Libero','Politecnico di Torino');
INSERT INTO Persona values
('VCYDLV34A25A761K','Matteo','Colombo',1111133336,'M','2002-07-
25','Vercelli','mcolombo9@polito.it','Chimica','Libero','Politecnico di Torino');
INSERT INTO Persona values
('HPTLCS61M19C057F','Angelo','Lombardo',1111133337,'M','1999-02-
24','Novara','alombardo1@polito.it','Chimica','Libero','Politecnico di Torino');
INSERT INTO Persona values
('PHGPAU98L18I747E','Greta','Colombo',1111133338,'F','1997-02-
13','Biella','gcolombo88@polito.it','Informarica','Libero','Politecnico di
Torino');
INSERT INTO Persona values
('PLCBDH40P25L755D','Chiara','Ricci',1111133339,'F','1998-03-
15','Torino','cricci@polito.it','Ingegneria Civile','Libero','Politecnico di
Torino');
INSERT INTO Persona values
('SLWBVR27D08C752K','Christiana','Vitali',1111133390,'F','2000-01-
12','Torino','cvitali@polito.it','Ingegneria Civile','Libero','Politecnico di
Torino');
INSERT INTO Persona values
('ZCPPMD67R46H475T','Domenica','Messina',1111133311,'F','2001-01-
23','Asti','dmessina@polito.it','Informatica','Libero','Politecnico di Torino');
```

```
INSERT INTO Persona values
('KVXTLD66L08D027L','Enrica','Bucci',1111133312,'F','2000-05-
22','Alessandria','ebucci@polito.it','Informatica','Libero','Politecnico di
Torino');

INSERT INTO Persona values
('PDKDCX73P45B179E','Stefano','Proietti',1000000009,'M','1982-02-
11','Torino','sproietti@polito.it','Informatica','Libero','Politecnico di Torino');
INSERT INTO Persona values
('VTRDJR62S23G136K','Marco','Boscolo',1000000010,'M','1983-01-
18','Milano','mboscolo@polito.it','Ingegneria Civile','Libero','Politecnico di
Torino');
INSERT INTO Persona values
('ZGCRRS55C11I991J','Francesca','Manca',1000000011,'F','1981-11-
20','Asti','fmanca@polito.it','Chimica','Libero','Politecnico di Torino');

--PERSONE Università degli studi di Firenze
INSERT INTO Persona values
('HHDNZP30A03L026W','Emilio','Sanna',1234577777,'M','2000-11-
22','Firenze','esanna@unifi.it','Filosofia','Libero','Università degli studi di
Firenze');
INSERT INTO Persona values
('GSRLBB44L63I673Q','Ernesto','Savastano',1234577778,'M','2001-10-
22','Grosseto','esavastano@unifi.it','Filosofia','Libero','Università degli studi
di Firenze');
INSERT INTO Persona values
('ZNGZLN84A26H250W','Samuele','Lallo',1234577779,'M','2002-11-
09','Pistoia','slallo@unifi.it','Lettere','Libero','Università degli studi di
Firenze');
INSERT INTO Persona values
('BRECHS46P55C327X','Fabio','Tarantino',1234577745,'M','1999-12-
02','Lucca','ftarantino@unifi.it','Lettere','Libero','Università degli studi di
Firenze');
INSERT INTO Persona values
('RWVCFL74S45D408R','Federico','Foscolo',1234577744,'M','1999-10-
12','Firenze','ffoscolo@unifi.it','Storia','Libero','Università degli studi di
Firenze');
INSERT INTO Persona values
('CYGFCR46E06G458T','Federica','Pavesi',1234577733,'F','1996-04-
21','Arezzo','fpavesi@unifi.it','Storia','Libero','Università degli studi di
Firenze');
INSERT INTO Persona values
('BWDXBM96P02G566J','Annamaria','Cerise',1234577722,'F','1995-08-
08','Arezzo','acerise@unifi.it','Storia','Libero','Università degli studi di
Firenze');
INSERT INTO Persona values
('ZMMCST86C27E915S','Gaia','Carraro',1234577711,'F','2001-06-
01','Roma','gcarraro@unifi.it','Sociologia','Libero','Università degli studi di
Firenze');
```

```
INSERT INTO Persona values
('DKODKC48P04L032G','Flavia','Gramsci',1234577734,'F','2000-04-
02','Pavia','fgramsci@unifi.it','Sociologia','Libero','Università degli studi di
Firenze');
INSERT INTO Persona values
('ZZCLHR28D55C147D','Flaviana','Greco',1234577789,'F','2000-03-
07','Padova','fgreco2@unifi.it','Sociologia','Libero','Università degli studi di
Firenze');

INSERT INTO Persona values
('PSJNVC64B67I749D','Luca','Prisco',1000000012,'M','1990-02-
14','Torino','lprisco@unifi.it','Sociologia','Libero','Università degli studi di
Firenze');
INSERT INTO Persona values
('GSGNRX97A26I982Y','Francesco','Enrisi',1000000013,'M','1986-01-
02','Milano','fenrisio@unifi.it','Lettere','Libero','Università degli studi di
Firenze');
INSERT INTO Persona values ('CVWCFW65L01I189Q','Chiara','Del
vecchio',1000000014,'F','1971-11-
07','Asti','cdelvecchio@unifi.it','Storia','Libero','Università degli studi di
Firenze');

--AULE
INSERT INTO Aula values (1, 'A01', 60, 'Ingegneria informatica', 'Università degli
studi di Salerno');
INSERT INTO Aula values (2, 'A02', 120, 'Fisica', 'Università degli studi di
Salerno');
INSERT INTO Aula values (3,'A03', 110, 'Matematica','Università degli studi di
Salerno');

INSERT INTO Aula values (4,'A04', 70 , 'Ingegneria chimica', 'Università degli
studi di Napoli');
INSERT INTO Aula values (5,'A05', 65 , 'Ingegneria Informatica', 'Università degli
studi di Napoli');
INSERT INTO Aula values (6,'A06', 65 , 'Economia', 'Università degli studi di
Napoli');

INSERT INTO Aula values (7, 'A07', 75, 'Ingegneria informatica', 'Politecnico di
Milano');
INSERT INTO Aula values (8, 'A08', 29, 'Fisica', 'Politecnico di Milano');
INSERT INTO Aula values (9,'A09', 10 , 'Matematica','Politecnico di Milano');

INSERT INTO Aula values (10,'A10', 100 , 'Ingegneria chimica', 'Politecnico di
Torino');
INSERT INTO Aula values (11,'A11', 160 , 'Ingegneria Informatica', 'Politecnico di
Torino');
INSERT INTO Aula values (12,'A12', 260 , 'Economia', 'Politecnico di Torino');
```

```
INSERT INTO Aula values (13,'A13', 260 , 'Sociologia', 'Università degli studi di
Firenze');
INSERT INTO Aula values (14,'A14', 202 , 'Lettere', 'Università degli studi di
Firenze');
INSERT INTO Aula values (15,'A15', 126 , 'Storia', 'Università degli studi di
Firenze');

-- PROFESSORE
INSERT INTO Professore values ('GTAMTT60L17H703S','Professore ordinario', 12);
INSERT INTO Professore values ('GRCGPP75D08H703K','Professore non ordinario', 25);
INSERT INTO Professore values ('MNZRNN70A48F839L','Professore ordinario', 56 );

INSERT INTO Professore values ('PACRCC60L17H703S','Professore ordinario', 11);
INSERT INTO Professore values ('MORGSP75D08H703K','Professore non ordinario', 22);
INSERT INTO Professore values ('MNCMNN70A48F839T','Professore ordinario', 55 );

INSERT INTO Professore values ('CSTPPY83B15E317O','Professore ordinario', 123);
INSERT INTO Professore values ('SNBZFC36C47H334A','Professore non ordinario', 145);
INSERT INTO Professore values ('WPKRHT53L24L325A','Professore ordinario', 156 );

INSERT INTO Professore values ('PDKDCX73P45B179E','Professore ordinario', 121);
INSERT INTO Professore values ('VTRDJR62S23G136K','Professore non ordinario', 252);
INSERT INTO Professore values ('ZGCRRS55C11I991J','Professore ordinario', 563 );

INSERT INTO Professore values ('PSJNVC64B67I749D','Professore ordinario', 89);
INSERT INTO Professore values ('GSGNRX97A26I982Y','Professore non ordinario', 4);
INSERT INTO Professore values ('CVWCFW65L01I189Q','Professore ordinario', 5 );

--STUDENTE
INSERT INTO Studente values ('TRNRMN00T52A783A', 1, 'Triennale', 2022);
INSERT INTO Studente values ('MLNMRC00D04C361Z', 2, 'Magistrale', 2021);
INSERT INTO Studente values ('VNDFNC00D68G964C', 3, 'Triennale', 2020);
INSERT INTO Studente values ('NTNGNN01B21A399J', 4, 'Ciclo unico', 2019);
INSERT INTO Studente values ('VTLLCU98B16F839G', 5, 'Ciclo unico', 2018);
INSERT INTO Studente values ('MNZLSE02L53B963F', 1, 'Triennale', 2022);
INSERT INTO Studente values ('DLCVCN97L05H703I', 2, 'Triennale', 2021);
INSERT INTO Studente values ('MRCMTT99L17A509P', 3, 'Ciclo unico', 2020);
INSERT INTO Studente values ('CSNRKE00C70F839Y', 4, 'Ciclo unico', 2019);
INSERT INTO Studente values ('CSCGAI00D67H703B', 5, 'Ciclo unico', 2018);

INSERT INTO Studente values ('RSSMTT00S12F839U', 1, 'Triennale', 2022);
INSERT INTO Studente values ('VRDMRC00S12F839T', 2, 'Magistrale', 2021);
INSERT INTO Studente values ('NREGPP00S12B990R', 3, 'Triennale', 2020);
INSERT INTO Studente values ('FRRLLN00S52A064B', 4, 'Ciclo unico', 2019);
INSERT INTO Studente values ('RSSFNC01S52F839H', 5, 'Ciclo unico', 2018);
INSERT INTO Studente values ('RMNDBR98A41B963Z', 1, 'Triennale', 2022);
INSERT INTO Studente values ('DLCLNR97A41B963J', 2, 'Triennale', 2021);
INSERT INTO Studente values ('FNTMTT99A01A509U', 3, 'Ciclo unico', 2020);
INSERT INTO Studente values ('CSTLCU00A01H703F', 4, 'Ciclo unico', 2019);
```

```
INSERT INTO Studente values ('DFLPLA00A01H703Q', 5, 'Ciclo unico', 2018);

INSERT INTO Studente values ('TFNHRD63R03B217K', 1, 'Triennale', 2022);
INSERT INTO Studente values ('MLKBZR68M42B350G', 2, 'Magistrale', 2021);
INSERT INTO Studente values ('ZTQTFR56A58L697L', 3, 'Triennale', 2020);
INSERT INTO Studente values ('SWFRYF37B65A948O', 4, 'Ciclo unico', 2019);
INSERT INTO Studente values ('QVTDNZ40L07C348V', 5, 'Ciclo unico', 2018);
INSERT INTO Studente values ('YMCBTX40A41I605Z', 1, 'Triennale', 2022);
INSERT INTO Studente values ('VFCZLG94R66M271D', 2, 'Triennale', 2021);
INSERT INTO Studente values ('RTGMTL30P44G691C', 3, 'Ciclo unico', 2020);
INSERT INTO Studente values ('TQBMNN79D44G997S', 4, 'Ciclo unico', 2019);
INSERT INTO Studente values ('MDMPTN91M02F527H', 5, 'Ciclo unico', 2018);

INSERT INTO Studente values ('ZDYFHO27M61G582Y', 1, 'Triennale', 2022);
INSERT INTO Studente values ('CSPDTM27E68E997W', 2, 'Magistrale', 2021);
INSERT INTO Studente values ('MJXHPZ53C69B591C', 3, 'Triennale', 2020);
INSERT INTO Studente values ('VCYDLV34A25A761K', 4, 'Ciclo unico', 2019);
INSERT INTO Studente values ('HPTLCS61M19C057F', 5, 'Ciclo unico', 2018);
INSERT INTO Studente values ('PHGPAU98L18I747E', 1, 'Triennale', 2022);
INSERT INTO Studente values ('PLCBDH40P25L755D', 2, 'Triennale', 2021);
INSERT INTO Studente values ('SLWBVR27D08C752K', 3, 'Ciclo unico', 2020);
INSERT INTO Studente values ('ZCPPMD67R46H475T', 4, 'Ciclo unico', 2019);
INSERT INTO Studente values ('KVXTLD66L08D027L', 5, 'Ciclo unico', 2018);

INSERT INTO Studente values ('HHDNZP30A03L026W', 1, 'Triennale', 2022);
INSERT INTO Studente values ('GSRLBB44L63I673Q', 2, 'Magistrale', 2021);
INSERT INTO Studente values ('ZNGZLN84A26H250W', 3, 'Triennale', 2020);
INSERT INTO Studente values ('BRECHS46P55C327X', 4, 'Ciclo unico', 2019);
INSERT INTO Studente values ('RWVCFL74S45D408R', 5, 'Ciclo unico', 2018);
INSERT INTO Studente values ('CYGFCR46E06G458T', 1, 'Triennale', 2022);
INSERT INTO Studente values ('BWDXBM96P02G566J', 2, 'Triennale', 2021);
INSERT INTO Studente values ('ZMMCST86C27E915S', 3, 'Ciclo unico', 2020);
INSERT INTO Studente values ('DKODKC48P04L032G', 4, 'Ciclo unico', 2019);
INSERT INTO Studente values ('ZZCLHR28D55C147D', 5, 'Ciclo unico', 2018);
COMMIT WORK;

--VACCINO
BEGIN TRANSACTION;
INSERT INTO Vaccino values ('Pfizer');
INSERT INTO Vaccino values ('Moderna');
INSERT INTO Vaccino values ('Vaxzevria');
INSERT INTO Vaccino values ('J&J');
COMMIT WORK;

--POSITIVI PASSATI
--UNIVERSITA' DI SALERNO
BEGIN TRANSACTION;

INSERT INTO Positivo values (1, false, true,'2022-03-01','TRNRMN00T52A783A' );
```

```
INSERT INTO Tampone values (default, '2022-02-28', '15:00', 'Positivo', 'Rapido',
1, null);
INSERT INTO Tampone values (default,'2022-03-
23','12:00','Negativo','Molecolare',1,null);
INSERT INTO Attestato_Positivo values(1,'Pfizer','2021-07-28');
INSERT INTO Attestato_Positivo values(1,'Vaxzevria','2022-02-23');

INSERT INTO Positivo values (2, false, false,'2022-03-27','MLNMRC00D04C361Z' );
INSERT INTO Tampone values (default, '2022-03-27', '18:00', 'Positivo', 'Rapido',
2, null);
INSERT INTO Tampone values (default,'2022-04-
07','20:00','Negativo','Molecolare',2,null);
INSERT INTO Attestato_Positivo values(2,'Moderna','2021-06-21');

INSERT INTO Positivo values (3, true, true,'2022-04-21','DLCVCN97L05H703I' );
INSERT INTO Tampone values (default, '2022-04-21', '08:00', 'Positivo',
'Molecolare', 3, null);
INSERT INTO Tampone values (default,'2022-04-
30','11:00','Negativo','Rapido',3,null);
INSERT INTO Attestato_Positivo values(3,'Pfizer','2022-01-03');
INSERT INTO Attestato_Positivo values(3,'J&J','2022-04-12');

INSERT INTO Positivo values (4, false, true,'2022-04-11','VNDFNC00D68G964C' );
INSERT INTO Tampone values (default, '2022-04-10', '10:00', 'Positivo',
'Molecolare', 4, null);
INSERT INTO Tampone values (default,'2022-04-
30','12:00','Negativo','Molecolare',4,null);
INSERT INTO Attestato_Positivo values(4,'Pfizer','2021-12-26');
INSERT INTO Attestato_Positivo values(4,'Pfizer','2022-03-13');

INSERT INTO Positivo values (5, false, false,'2022-05-05','NTNGNN01B21A399J' );
INSERT INTO Tampone values (default, '2022-05-05', '15:00', 'Positivo', 'Rapido',
5, null);
INSERT INTO Tampone values (default,'2022-05-
21','12:00','Negativo','Rapido',5,null);
INSERT INTO Attestato_Positivo values(5,'Pfizer','2021-02-17');
INSERT INTO Attestato_Positivo values(5,'Pfizer','2021-06-10');
INSERT INTO Attestato_Positivo values(5,'Vaxzevria','2022-04-04');

INSERT INTO Positivo values (6, false, true,'2022-05-29','VTLLCU98B16F839G' );
INSERT INTO Tampone values (default, '2022-05-27', '15:00', 'Positivo', 'Rapido',
6, null);
INSERT INTO Tampone values (default,'2022-06-
09','12:00','Negativo','Molecolare',6,null);


INSERT INTO Positivo values (7, false, false,'2022-04-05','GTAMTT60L17H703S' );
INSERT INTO Tampone values (default, '2022-04-05', '20:00', 'Positivo',
'Molecolare', 7, null);
```

```
INSERT INTO Tampone values (default,'2022-04-
13','19:00','Negativo','Rapido',7,null);
INSERT INTO Attestato_Positivo values(7,'Vaxzevria','2022-02-17');
INSERT INTO Attestato_Positivo values(7,'Vaxzevria','2022-04-21');


INSERT INTO Positivo values (8, false, false,'2022-02-02','MNZRNN70A48F839L' );
INSERT INTO Tampone values (default, '2022-02-01', '08:00', 'Positivo', 'Rapido',
8, null);
INSERT INTO Tampone values (default,'2022-03-
02','11:00','Negativo','Rapido',8,null);
INSERT INTO Attestato_Positivo values(8,'Pfizer','2021-08-09');
INSERT INTO Attestato_Positivo values(8,'J&J','2021-12-11');

-- UNIVERSITA' DI NAPOLI
INSERT INTO Positivo values (9, false, true,'2022-03-11','RSSMTT00S12F839U' );
INSERT INTO Tampone values (default, '2022-03-11', '13:00', 'Positivo', 'Rapido',
9, null);
INSERT INTO Tampone values (default,'2022-03-
26','12:00','Negativo','Molecolare',9,null);
INSERT INTO Attestato_Positivo values(9,'J&J','2021-03-28');
INSERT INTO Attestato_Positivo values(9,'J&J','2021-12-23');

INSERT INTO Positivo values (10, true, true,'2022-03-25','VRDMRC00S12F839T' );
INSERT INTO Tampone values (default, '2022-03-23', '18:00', 'Positivo', 'Rapido',
10, null);
INSERT INTO Tampone values (default,'2022-04-
08','20:00','Negativo','Rapido',10,null);
INSERT INTO Attestato_Positivo values(10,'Moderna','2021-09-29');

INSERT INTO Positivo values (11, true, true,'2022-03-21','NREGPP00S12B990R' );
INSERT INTO Tampone values (default, '2022-03-21', '08:00', 'Positivo',
'Molecolare', 11, null);
INSERT INTO Tampone values (default,'2022-04-
01','11:00','Negativo','Rapido',11,null);
INSERT INTO Attestato_Positivo values(11,'Moderna','2022-02-13');
INSERT INTO Attestato_Positivo values(11,'J&J','2022-03-13');

INSERT INTO Positivo values (12, false, true,'2022-04-20','FRRLLN00S52A064B' );
INSERT INTO Tampone values (default, '2022-04-20', '10:00', 'Positivo',
'Molecolare', 12, null);
INSERT INTO Tampone values (default,'2022-04-
29','12:00','Negativo','Molecolare',12,null);
INSERT INTO Attestato_Positivo values(12,'Moderna','2021-11-30');
INSERT INTO Attestato_Positivo values(12,'Moderna','2022-02-21');

INSERT INTO Positivo values (13, false, true,'2022-04-30','RSSFNC01S52F839H' );
INSERT INTO Tampone values (default, '2022-04-30', '18:00', 'Positivo', 'Rapido',
13, null);
```

```
INSERT INTO Tampone values (default,'2022-05-
13','18:00','Negativo','Rapido',13,null);
INSERT INTO Attestato_Positivo values(13,'Vaxzevria','2021-04-17');
INSERT INTO Attestato_Positivo values(13,'Vaxzevria','2021-07-19');
INSERT INTO Attestato_Positivo values(13,'Vaxzevria','2022-01-09');

INSERT INTO Positivo values (14, false, true,'2022-05-15','RMNDBR98A41B963Z' );
INSERT INTO Tampone values (default, '2022-05-14', '15:00', 'Positivo', 'Rapido',
14, null);
INSERT INTO Tampone values (default,'2022-05-
30','12:00','Negativo','Molecolare',14,null);
INSERT INTO Attestato_Positivo values(14,'Vaxzevria','2022-05-05');


INSERT INTO Positivo values (15, false, false,'2022-04-05','MORGSP75D08H703K' );
INSERT INTO Tampone values (default, '2022-04-05', '20:00', 'Positivo',
'Molecolare', 15, null);
INSERT INTO Tampone values (default,'2022-04-
13','19:00','Negativo','Rapido',15,null);
INSERT INTO Attestato_Positivo values(15,'Vaxzevria','2021-12-11');
INSERT INTO Attestato_Positivo values(15,'Vaxzevria','2022-03-22');


INSERT INTO Positivo values (16, false, false,'2022-02-02','PACRCC60L17H703S' );
INSERT INTO Tampone values (default, '2022-02-01', '08:00', 'Positivo', 'Rapido',
16, null);
INSERT INTO Tampone values (default,'2022-02-
11','11:00','Negativo','Rapido',16,null);
INSERT INTO Attestato_Positivo values(16,'Pfizer','2021-04-19');
INSERT INTO Attestato_Positivo values(16,'J&J','2021-09-12');


-- UNIVERSITA' DI MILANO
INSERT INTO Positivo values (17, false, true,'2022-02-18','ZTQTFR56A58L697L' );
INSERT INTO Tampone values (default, '2022-02-17', '13:00', 'Positivo', 'Rapido',
17, null);
INSERT INTO Tampone values (default,'2022-03-
09','12:00','Negativo','Molecolare',17,null);
INSERT INTO Attestato_Positivo values(17,'Vaxzevria','2021-10-28');
INSERT INTO Attestato_Positivo values(17,'J&J','2022-02-01');

INSERT INTO Positivo values (18, true, true,'2022-03-02','VFCZLG94R66M271D' );
INSERT INTO Tampone values (default, '2022-03-02', '18:00', 'Positivo', 'Rapido',
18, null);
INSERT INTO Tampone values (default,'2022-03-
30','20:00','Negativo','Rapido',18,null);
INSERT INTO Attestato_Positivo values(18,'Vaxzevria','2022-01-29');

INSERT INTO Positivo values (19, true, true,'2022-04-10','MDMPTN91M02F527H' );
```

```
INSERT INTO Tampone values (default, '2022-04-10', '08:00', 'Positivo',
'Molecolare', 19, null);
INSERT INTO Tampone values (default,'2022-04-
21','11:00','Negativo','Rapido',19,null);
INSERT INTO Attestato_Positivo values(19,'Pfizer','2022-01-10');
INSERT INTO Attestato_Positivo values(19,'J&J','2022-04-03');

INSERT INTO Positivo values (20, false, true,'2022-04-21','MLKBZR68M42B350G' );
INSERT INTO Tampone values (default, '2022-04-20', '10:00', 'Positivo',
'Molecolare', 20, null);
INSERT INTO Tampone values (default,'2022-05-
15','12:00','Negativo','Molecolare',20,null);
INSERT INTO Attestato_Positivo values(20,'Moderna','2021-11-30');
INSERT INTO Attestato_Positivo values(20,'Pfizer','2022-03-11');

INSERT INTO Positivo values (21, false, true,'2022-04-30','TFNHRD63R03B217K' );
INSERT INTO Tampone values (default, '2022-04-30', '18:00', 'Positivo', 'Rapido',
21, null);
INSERT INTO Tampone values (default,'2022-05-
22','18:00','Negativo','Rapido',21,null);
INSERT INTO Attestato_Positivo values(21,'Pfizer','2021-09-07');
INSERT INTO Attestato_Positivo values(21,'Vaxzevria','2021-11-09');
INSERT INTO Attestato_Positivo values(21,'Vaxzevria','2022-03-29');

INSERT INTO Positivo values (22, false, true,'2022-05-12','QVTDNZ40L07C348V' );
INSERT INTO Tampone values (default, '2022-05-12', '15:00', 'Positivo', 'Rapido',
22, null);
INSERT INTO Tampone values (default,'2022-06-
01','12:00','Negativo','Molecolare',22,null);
INSERT INTO Attestato_Positivo values(22,'Pfizer','2022-04-16');


INSERT INTO Positivo values (23, false, false,'2022-05-10','CSTPPY83B15E317O' );
INSERT INTO Tampone values (default, '2022-05-07', '20:00', 'Positivo',
'Molecolare', 23, null);
INSERT INTO Tampone values (default,'2022-05-
28','19:00','Negativo','Rapido',23,null);
INSERT INTO Attestato_Positivo values(23,'Moderna','2021-06-16');
INSERT INTO Attestato_Positivo values(23,'Vaxzevria','2022-04-26');


INSERT INTO Positivo values (24, false, false,'2022-01-23','SNBZFC36C47H334A' );
INSERT INTO Tampone values (default, '2022-01-23', '08:00', 'Positivo', 'Rapido',
24, null);
INSERT INTO Tampone values (default,'2022-02-
19','11:00','Negativo','Rapido',24,null);
INSERT INTO Attestato_Positivo values(24,'Pfizer','2021-04-11');
INSERT INTO Attestato_Positivo values(24,'Pfizer','2021-09-19');
```

```
-- UNIVERSITA' DI FIRENZE

INSERT INTO Positivo values (25, false, false,'2022-01-31','ZNGZLN84A26H250W' );
INSERT INTO Tampone values (default, '2022-01-31', '19:00', 'Positivo', 'Rapido',
25, null);
INSERT INTO Tampone values (default,'2022-02-
09','12:00','Negativo','Molecolare',25,null);
INSERT INTO Attestato_Positivo values(25,'Vaxzevria','2021-11-18');
INSERT INTO Attestato_Positivo values(25,'Vaxzevria','2022-01-24');

INSERT INTO Positivo values (26, false, false,'2022-01-31','GSRLBB44L63I673Q' );
INSERT INTO Tampone values (default, '2022-01-30', '14:00', 'Positivo', 'Rapido',
26, null);
INSERT INTO Tampone values (default,'2022-02-
07','18:00','Negativo','Rapido',26,null);
INSERT INTO Attestato_Positivo values(26,'J&J','2021-12-05');

INSERT INTO Positivo values (27, false, false,'2022-03-21','ZMMCST86C27E915S' );
INSERT INTO Tampone values (default, '2022-03-21', '08:00', 'Positivo',
'Molecolare', 27, null);
INSERT INTO Tampone values (default,'2022-04-
01','11:00','Negativo','Rapido',27,null);
INSERT INTO Attestato_Positivo values(27,'Pfizer','2022-01-19');
INSERT INTO Attestato_Positivo values(27,'Pfizer','2022-03-02');

INSERT INTO Positivo values (28, false, true,'2022-04-17','HHDNZP30A03L026W' );
INSERT INTO Tampone values (default, '2022-04-16', '18:00', 'Positivo',
'Molecolare', 28, null);
INSERT INTO Tampone values (default,'2022-04-
26','17:00','Negativo','Molecolare',28,null);
INSERT INTO Attestato_Positivo values(28,'Pfizer','2021-10-03');
INSERT INTO Attestato_Positivo values(28,'Pfizer','2022-04-08');

INSERT INTO Positivo values (29, false, false,'2022-04-20','DKODKC48P04L032G' );
INSERT INTO Tampone values (default, '2022-04-20', '08:00', 'Positivo', 'Rapido',
29, null);
INSERT INTO Tampone values (default,'2022-04-
30','17:00','Negativo','Rapido',29,null);
INSERT INTO Attestato_Positivo values(29,'Pfizer','2020-12-17');
INSERT INTO Attestato_Positivo values(29,'Moderna','2021-06-12');
INSERT INTO Attestato_Positivo values(29,'J&J','2022-04-07');

INSERT INTO Positivo values (30, true, true,'2022-05-02','ZZCLHR28D55C147D' );
INSERT INTO Tampone values (default, '2022-05-01', '15:00', 'Positivo', 'Rapido',
30, null);
INSERT INTO Tampone values (default,'2022-05-
07','12:00','Negativo','Molecolare',30,null);
INSERT INTO Attestato_Positivo values(30,'Pfizer','2022-04-19');
```

```
INSERT INTO Positivo values (31, false, false,'2022-05-19','CVWCFW65L01I189Q' );
INSERT INTO Tampone values (default, '2022-05-18', '20:00', 'Positivo',
'Molecolare', 31, null);
INSERT INTO Tampone values (default,'2022-05-
25','19:00','Negativo','Rapido',31,null);
INSERT INTO Attestato_Positivo values(31,'Moderna','2021-06-20');
INSERT INTO Attestato_Positivo values(31,'Moderna','2022-04-27');


INSERT INTO Positivo values (32, false, true,'2022-04-13','GSGNRX97A26I982Y' );
INSERT INTO Tampone values (default, '2022-04-13', '08:00', 'Positivo', 'Rapido',
32, null);
INSERT INTO Tampone values (default,'2022-04-
24','11:00','Negativo','Rapido',32,null);
INSERT INTO Attestato_Positivo values(32,'J&J','2021-09-11');
INSERT INTO Attestato_Positivo values(32,'Pfizer','2021-11-19');

-- UNIVERSITA' DI TORINO
INSERT INTO Positivo values (33, false, false,'2022-01-10','MJXHPZ53C69B591C' );
INSERT INTO Tampone values (default, '2022-01-10', '09:00', 'Positivo', 'Rapido',
33, null);
INSERT INTO Tampone values (default,'2022-02-
07','17:00','Negativo','Molecolare',33,null);
INSERT INTO Attestato_Positivo values(33,'Vaxzevria','2021-10-28');
INSERT INTO Attestato_Positivo values(33,'Vaxzevria','2021-12-24');

INSERT INTO Positivo values (34, false, false,'2022-01-28','VCYDLV34A25A761K' );
INSERT INTO Tampone values (default, '2022-01-28', '12:00', 'Positivo', 'Rapido',
34, null);
INSERT INTO Tampone values (default,'2022-02-
17','14:00','Negativo','Rapido',34,null);
INSERT INTO Attestato_Positivo values(34,'Pfizer','2021-01-15');

INSERT INTO Positivo values (35, false, false,'2022-03-11','CSPDTM27E68E997W' );
INSERT INTO Tampone values (default, '2022-03-11', '18:00', 'Positivo',
'Molecolare', 35, null);
INSERT INTO Tampone values (default,'2022-04-
12','14:00','Negativo','Rapido',35,null);
INSERT INTO Attestato_Positivo values(35,'Pfizer','2022-01-25');
INSERT INTO Attestato_Positivo values(35,'Vaxzevria','2022-03-09');

INSERT INTO Positivo values (36, false, true,'2022-04-27','ZCPPMD67R46H475T' );
INSERT INTO Tampone values (default, '2022-04-26', '20:00', 'Positivo',
'Molecolare', 36, null);
INSERT INTO Tampone values (default,'2022-05-
16','18:00','Negativo','Molecolare',36,null);
INSERT INTO Attestato_Positivo values(36,'Pfizer','2021-09-23');
INSERT INTO Attestato_Positivo values(36,'Pfizer','2022-04-07');
```

```
INSERT INTO Positivo values (37, false, false,'2022-04-10','ZDYFHO27M61G582Y' );
INSERT INTO Tampone values (default, '2022-04-07', '18:00', 'Positivo', 'Rapido',
37, null);
INSERT INTO Tampone values (default,'2022-05-
12','19:00','Negativo','Rapido',37,null);
INSERT INTO Attestato_Positivo values(37,'Pfizer','2021-04-23');
INSERT INTO Attestato_Positivo values(37,'Moderna','2021-06-27');
INSERT INTO Attestato_Positivo values(37,'Moderna','2022-04-01');

INSERT INTO Positivo values (38, true, true,'2022-05-12','KVXTLD66L08D027L' );
INSERT INTO Tampone values (default, '2022-05-11', '07:30', 'Positivo', 'Rapido',
38, null);
INSERT INTO Tampone values (default,'2022-05-
19','16:00','Negativo','Molecolare',38,null);
INSERT INTO Attestato_Positivo values(38,'Pfizer','2022-04-29');


INSERT INTO Positivo values (39, false, false,'2022-05-09','ZGCRRS55C11I991J' );
INSERT INTO Tampone values (default, '2022-05-08', '20:00', 'Positivo',
'Molecolare', 39, null);
INSERT INTO Tampone values (default,'2022-06-
01','19:00','Negativo','Rapido',39,null);
INSERT INTO Attestato_Positivo values(39,'J&J','2021-04-20');
INSERT INTO Attestato_Positivo values(39,'Moderna','2022-04-20');


INSERT INTO Positivo values (40, false, true,'2022-03-03','PDKDCX73P45B179E' );
INSERT INTO Tampone values (default, '2022-03-01', '08:00', 'Positivo', 'Rapido',
40, null);
INSERT INTO Tampone values (default,'2022-04-
02','11:00','Negativo','Rapido',40,null);
INSERT INTO Attestato_Positivo values(40,'J&J','2021-09-19');
INSERT INTO Attestato_Positivo values(40,'J&J','2022-02-28');
COMMIT WORK;


-- A questo punto è possibile verificare le query di Media giorni positività per
Regione e la Ricerca
--vaccinati con due dosi e positivi per più di 21 giorni.

/* POPOLAMENTO CALENDARIO PER MOSTRARE AULE LIBERE E OCCUPATE
Per il nostro database abbiamo previsto l'inserimento di Attività che siano
conseguenti al momento
in cui le si inserisce all'interno del database. Per questo motivo inseriamo alcune
attività schedulate
nel calendario che occupino interamente alcune giornate, in modo tale che possiate
controllare
la query delle aule inserendo attività nel giorno in cui state controllando.
Inoltre ricordiamo che
```

```
in seguito all'inserimento e al tracciamento dei contatti alcune aule risulteranno
libere
poichè l'attività viene svolta a distanza quando il professore risulta positivo*/
BEGIN TRANSACTION;

INSERT INTO Calendario values(1,'07:00','22:00','Lunedì','Programmazione',2,1);
INSERT INTO Calendario values(2,'07:00','22:00','Martedì','Fisica',2,2);
INSERT INTO Calendario values(3,'07:00','22:00','Mercoledì','Matematica',2,1);
INSERT INTO Calendario values(4,'07:00','22:00','Giovedì','Basi di Dati',2,3);
INSERT INTO Calendario values(5,'07:00','22:00','Venerdì','IoT',2,3);

INSERT INTO Calendario values(6,'07:00','22:00','Lunedì','Programmazione',2,4);
INSERT INTO Calendario values(7,'07:00','22:00','Martedì','Fisica',2,4);
INSERT INTO Calendario values(8,'07:00','22:00','Mercoledì','Matematica',2,6);
INSERT INTO Calendario values(9,'07:00','22:00','Giovedì','Basi di Dati',2,5);
INSERT INTO Calendario values(10,'07:00','22:00','Venerdì','IoT',2,6);

INSERT INTO Calendario values(11,'07:00','22:00','Lunedì','Programmazione',2,7);
INSERT INTO Calendario values(12,'07:00','22:00','Martedì','Fisica',2,7);
INSERT INTO Calendario values(13,'07:00','22:00','Mercoledì','Matematica',2,7);
INSERT INTO Calendario values(14,'07:00','22:00','Giovedì','Basi di Dati',2,9);
INSERT INTO Calendario values(15,'07:00','22:00','Venerdì','IoT',2,8);

INSERT INTO Calendario values(16,'07:00','22:00','Lunedì','Programmazione',2,10);
INSERT INTO Calendario values(17,'07:00','22:00','Martedì','Fisica',2,11);
INSERT INTO Calendario values(18,'07:00','22:00','Mercoledì','Matematica',2,11);
INSERT INTO Calendario values(19,'07:00','22:00','Giovedì','Basi di Dati',2,11);
INSERT INTO Calendario values(20,'07:00','22:00','Venerdì','IoT',2,12);

INSERT INTO Calendario values(21,'07:00','22:00','Lunedì','Programmazione',2,13);
INSERT INTO Calendario values(22,'07:00','22:00','Martedì','Fisica',2,13);
INSERT INTO Calendario values(23,'07:00','22:00','Mercoledì','Matematica',2,14);
INSERT INTO Calendario values(24,'07:00','22:00','Giovedì','Basi di Dati',2,14);
INSERT INTO Calendario values(25,'07:00','22:00','Venerdì','IoT',2,15);
COMMIT WORK;

-- A seconda del giorno della settimana runnare uno dei seguenti statement per
inserire le attività
--nel giorno corretto, poi eseguire la query per la ricerca delle aule libere

-- Se è lunedì
BEGIN TRANSACTION;

INSERT INTO Attività values(CURRENT_DATE,1,DEFAULT,20,60,'GTAMTT60L17H703S');
INSERT INTO Attività values(CURRENT_DATE,6,DEFAULT,25,70,'PACRCC60L17H703S');
INSERT INTO Attività values(CURRENT_DATE,11,DEFAULT,25,75,'CSTPPY83B15E317O');
INSERT INTO Attività values(CURRENT_DATE,16,DEFAULT,50,100,'PDKDCX73P45B179E');
INSERT INTO Attività values(CURRENT_DATE,21,DEFAULT,100,260,'CVWCFW65L01I189Q');
COMMIT WORK;
```

```
-- Se è martedì
BEGIN TRANSACTION;

INSERT INTO Attività values(CURRENT_DATE,2,DEFAULT,60,120,'GTAMTT60L17H703S');
INSERT INTO Attività values(CURRENT_DATE,7,DEFAULT,25,70,'PACRCC60L17H703S');
INSERT INTO Attività values(CURRENT_DATE,12,DEFAULT,45,75,'CSTPPY83B15E317O');
INSERT INTO Attività values(CURRENT_DATE,17,DEFAULT,100,160,'PDKDCX73P45B179E');
INSERT INTO Attività values(CURRENT_DATE,22,DEFAULT,80,260,'CVWCFW65L01I189Q');
COMMIT WORK;

-- Se è mercoledì
BEGIN TRANSACTION;

INSERT INTO Attività values(CURRENT_DATE,3,DEFAULT,60,60,'GRCGPP75D08H703K');
INSERT INTO Attività values(CURRENT_DATE,8,DEFAULT,65,65,'MORGSP75D08H703K');
INSERT INTO Attività values(CURRENT_DATE,13,DEFAULT,15,75,'CSTPPY83B15E317O');
INSERT INTO Attività values(CURRENT_DATE,18,DEFAULT,50,160,'PDKDCX73P45B179E');
INSERT INTO Attività values(CURRENT_DATE,23,DEFAULT,101,202,'CVWCFW65L01I189Q');
COMMIT WORK;

-- Se è giovedì
BEGIN TRANSACTION;

INSERT INTO Attività values(CURRENT_DATE,4,DEFAULT,60,110,'GRCGPP75D08H703K');
INSERT INTO Attività values(CURRENT_DATE,9,DEFAULT,65,65,'MORGSP75D08H703K');
INSERT INTO Attività values(CURRENT_DATE,14,DEFAULT,10,10,'CSTPPY83B15E317O');
INSERT INTO Attività values(CURRENT_DATE,19,DEFAULT,100,160,'PDKDCX73P45B179E');
INSERT INTO Attività values(CURRENT_DATE,24,DEFAULT,101,202,'CVWCFW65L01I189Q');
COMMIT WORK;

--Se è venerdì
BEGIN TRANSACTION;

INSERT INTO Attività values(CURRENT_DATE,5,DEFAULT,55,110,'MNZRNN70A48F839L');
INSERT INTO Attività values(CURRENT_DATE,10,DEFAULT,50,65,'MNCMNN70A48F839T');
INSERT INTO Attività values(CURRENT_DATE,15,DEFAULT,20,29,'CSTPPY83B15E317O');
INSERT INTO Attività values(CURRENT_DATE,20,DEFAULT,130,260,'PDKDCX73P45B179E');
INSERT INTO Attività values(CURRENT_DATE,25,DEFAULT,113,126,'CVWCFW65L01I189Q');
COMMIT WORK;


/* Per lo stesso motivo, la query per il tracciamento dei contatti dei positivi
odierni, lavoriamo allo stesso
modo, inserendo prenotazioni relative alle attività, in base al giorno in cui state
verificando, per semplicità
inseriamo prenotazioni relative soltanto all'Università degli studi di Salerno e al
Politecnico di Milano,
anche per rendere gli statement più leggibili*/
```

```
--Se è lunedì
BEGIN TRANSACTION;

INSERT INTO Prenotazione VALUES ('TRNRMN00T52A783A',CURRENT_DATE,1);
INSERT INTO Prenotazione VALUES ('MLNMRC00D04C361Z',CURRENT_DATE,1);
INSERT INTO Prenotazione VALUES ('VNDFNC00D68G964C',CURRENT_DATE,1);
INSERT INTO Prenotazione VALUES ('NTNGNN01B21A399J',CURRENT_DATE,1);

INSERT INTO Prenotazione VALUES ('MLKBZR68M42B350G',CURRENT_DATE,11);
INSERT INTO Prenotazione VALUES ('ZTQTFR56A58L697L',CURRENT_DATE,11);
INSERT INTO Prenotazione VALUES ('SWFRYF37B65A948O',CURRENT_DATE,11);
INSERT INTO Prenotazione VALUES ('QVTDNZ40L07C348V',CURRENT_DATE,11);
COMMIT WORK;

--Se è martedì
BEGIN TRANSACTION;

INSERT INTO Prenotazione VALUES ('TRNRMN00T52A783A',CURRENT_DATE,2);
INSERT INTO Prenotazione VALUES ('MLNMRC00D04C361Z',CURRENT_DATE,2);
INSERT INTO Prenotazione VALUES ('VNDFNC00D68G964C',CURRENT_DATE,2);
INSERT INTO Prenotazione VALUES ('NTNGNN01B21A399J',CURRENT_DATE,2);

INSERT INTO Prenotazione VALUES ('MLKBZR68M42B350G',CURRENT_DATE,12);
INSERT INTO Prenotazione VALUES ('ZTQTFR56A58L697L',CURRENT_DATE,12);
INSERT INTO Prenotazione VALUES ('SWFRYF37B65A948O',CURRENT_DATE,12);
INSERT INTO Prenotazione VALUES ('QVTDNZ40L07C348V',CURRENT_DATE,12);
COMMIT WORK;

-- Se è mercoledì
BEGIN TRANSACTION;

INSERT INTO Prenotazione VALUES ('TRNRMN00T52A783A',CURRENT_DATE,3);
INSERT INTO Prenotazione VALUES ('MLNMRC00D04C361Z',CURRENT_DATE,3);
INSERT INTO Prenotazione VALUES ('VNDFNC00D68G964C',CURRENT_DATE,3);
INSERT INTO Prenotazione VALUES ('NTNGNN01B21A399J',CURRENT_DATE,3);

INSERT INTO Prenotazione VALUES ('MLKBZR68M42B350G',CURRENT_DATE,13);
INSERT INTO Prenotazione VALUES ('ZTQTFR56A58L697L',CURRENT_DATE,13);
INSERT INTO Prenotazione VALUES ('SWFRYF37B65A948O',CURRENT_DATE,13);
INSERT INTO Prenotazione VALUES ('QVTDNZ40L07C348V',CURRENT_DATE,13);
COMMIT WORK;

-- Se è giovedì
BEGIN TRANSACTION;

INSERT INTO Prenotazione VALUES ('TRNRMN00T52A783A',CURRENT_DATE,4);
INSERT INTO Prenotazione VALUES ('MLNMRC00D04C361Z',CURRENT_DATE,4);
INSERT INTO Prenotazione VALUES ('VNDFNC00D68G964C',CURRENT_DATE,4);
```

```
INSERT INTO Prenotazione VALUES ('NTNGNN01B21A399J',CURRENT_DATE,4);

INSERT INTO Prenotazione VALUES ('MLKBZR68M42B350G',CURRENT_DATE,14);
INSERT INTO Prenotazione VALUES ('ZTQTFR56A58L697L',CURRENT_DATE,14);
INSERT INTO Prenotazione VALUES ('SWFRYF37B65A948O',CURRENT_DATE,14);
INSERT INTO Prenotazione VALUES ('QVTDNZ40L07C348V',CURRENT_DATE,14);
COMMIT WORK;

-- Se è venerdì
BEGIN TRANSACTION;

INSERT INTO Prenotazione VALUES ('TRNRMN00T52A783A',CURRENT_DATE,5);
INSERT INTO Prenotazione VALUES ('MLNMRC00D04C361Z',CURRENT_DATE,5);
INSERT INTO Prenotazione VALUES ('VNDFNC00D68G964C',CURRENT_DATE,5);
INSERT INTO Prenotazione VALUES ('NTNGNN01B21A399J',CURRENT_DATE,5);

INSERT INTO Prenotazione VALUES ('MLKBZR68M42B350G',CURRENT_DATE,15);
INSERT INTO Prenotazione VALUES ('ZTQTFR56A58L697L',CURRENT_DATE,15);
INSERT INTO Prenotazione VALUES ('SWFRYF37B65A948O',CURRENT_DATE,15);
INSERT INTO Prenotazione VALUES ('QVTDNZ40L07C348V',CURRENT_DATE,15);
COMMIT WORK;



/* Dopodichè rendiamo positivo per l'università di salerno uno studente
dell'attività, mentre per
il politecnico rendiamo positivo uno studente e un professore così da poter
effettuare la query per il tracciamento
dei contatti */
BEGIN TRANSACTION;

INSERT INTO Positivo values (41, false, false,CURRENT_DATE,'NTNGNN01B21A399J' );
INSERT INTO Tampone values (default, CURRENT_DATE, CURRENT_TIME, 'Positivo',
'Molecolare', 41, null);

INSERT INTO Positivo values (42, false, false,CURRENT_DATE,'CSTPPY83B15E317O' );
INSERT INTO Tampone values (default, CURRENT_DATE, CURRENT_TIME, 'Positivo',
'Molecolare', 42, null);

INSERT INTO Positivo values (43, false, false,CURRENT_DATE,'QVTDNZ40L07C348V' );
INSERT INTO Tampone values (default, CURRENT_DATE, CURRENT_TIME, 'Positivo',
'Molecolare', 43, null);
COMMIT WORK;

/* Una volta effettuata la query potremmo anche popolare la tabella di Attestato
Contatto inserendo le
relative date di somministrazione dei vaccini dei vari contatti */
```

```
-- POSITIVI AL MOMENTO
BEGIN TRANSACTION;

INSERT INTO Positivo values (44, false, false,'2022-06-02','CSCGAI00D67H703B' );
INSERT INTO Tampone values (default,'2022-06-02','10:00' , 'Positivo',
'Molecolare', 44, null);
INSERT INTO Positivo values (45, false, false,'2022-06-05','CSNRKE00C70F839Y' );
INSERT INTO Tampone values (default,'2022-06-04','10:00' , 'Positivo',
'Molecolare', 45, null);
INSERT INTO Positivo values (46, false, false,'2022-06-05','MRCMTT99L17A509P' );
INSERT INTO Tampone values (default,'2022-05-28','12:00' , 'Positivo',
'Molecolare', 46, null);
INSERT INTO Positivo values (47, false, false,'2022-06-05','DLCVCN97L05H703I' );
INSERT INTO Tampone values (default,'2022-05-30','12:00' , 'Positivo',
'Molecolare', 47, null);


INSERT INTO Positivo values (48, false, false,'2022-06-02','DFLPLA00A01H703Q' );
INSERT INTO Tampone values (default,'2022-06-01','14:00' , 'Positivo',
'Molecolare', 48, null);


INSERT INTO Positivo values (49, false, false,'2022-06-04','YMCBTX40A41I605Z' );
INSERT INTO Tampone values (default,'2022-06-03','17:00' , 'Positivo',
'Molecolare', 49, null);
INSERT INTO Positivo values (50, false, false,'2022-05-25','VFCZLG94R66M271D' );
INSERT INTO Tampone values (default,'2022-05-25','15:00' , 'Positivo',
'Molecolare', 50, null);
INSERT INTO Positivo values (51, false, false,'2022-06-08','RTGMTL30P44G691C' );
INSERT INTO Tampone values (default,'2022-06-06','12:00' , 'Positivo',
'Molecolare', 51, null);
INSERT INTO Positivo values (52, false, false,'2022-06-05','TQBMNN79D44G997S' );
INSERT INTO Tampone values (default,'2022-06-05','13:00' , 'Positivo',
'Molecolare', 52, null);
INSERT INTO Positivo values (53, false, false,'2022-06-01','MDMPTN91M02F527H' );
INSERT INTO Tampone values (default,'2022-05-31','14:00' , 'Positivo',
'Molecolare', 53, null);

INSERT INTO Positivo values (54, false, false,'2022-06-04','ZZCLHR28D55C147D' );
INSERT INTO Tampone values (default,'2022-06-03','17:00' , 'Positivo',
'Molecolare', 54, null);
INSERT INTO Positivo values (55, false, false,'2022-05-25','DKODKC48P04L032G' );
INSERT INTO Tampone values (default,'2022-05-25','15:00' , 'Positivo',
'Molecolare', 55, null);
INSERT INTO Positivo values (56, false, false,'2022-06-08','ZMMCST86C27E915S' );
INSERT INTO Tampone values (default,'2022-06-06','12:00' , 'Positivo',
'Molecolare', 56, null);
INSERT INTO Positivo values (57, false, false,'2022-06-05','BWDXBM96P02G566J' );
```

```
INSERT INTO Tampone values (default,'2022-06-05','13:00' , 'Positivo',
'Molecolare', 57, null);
INSERT INTO Positivo values (58, false, false,'2022-06-01','CYGFCR46E06G458T' );
INSERT INTO Tampone values (default,'2022-05-31','14:00' , 'Positivo',
'Molecolare', 58, null);
INSERT INTO Positivo values (59, false, false,'2022-06-05','RWVCFL74S45D408R' );
INSERT INTO Tampone values (default,'2022-06-05','13:00' , 'Positivo',
'Molecolare', 59, null);
INSERT INTO Positivo values (60, false, false,'2022-06-01','ZNGZLN84A26H250W' );
INSERT INTO Tampone values (default,'2022-05-31','14:00' , 'Positivo',
'Molecolare', 60, null);
INSERT INTO Positivo values (61, false, false,'2022-06-01','PSJNVC64B67I749D' );
INSERT INTO Tampone values (default,'2022-05-31','14:00' , 'Positivo',
'Molecolare', 61, null);

INSERT INTO Positivo values (62, false, false,'2022-06-05','ZDYFHO27M61G582Y' );
INSERT INTO Tampone values (default,'2022-06-05','13:00' , 'Positivo',
'Molecolare', 62, null);
INSERT INTO Positivo values (63, false, false,'2022-06-01','MJXHPZ53C69B591C' );
INSERT INTO Tampone values (default,'2022-05-31','14:00' , 'Positivo',
'Molecolare', 63, null);
COMMIT WORK;
```

7. Query SQL

Workpackage	*Task*	*Responsabile*
WP3	SQL: Query	Tarantino Ramona

7.1. Query con operatore di aggregazione e join: Media giorni positività per Regione

La query seleziona, per ogni regione in cui è presente almeno un'università, la media di giorni di positività degli studenti o professori risultati positivi al covid:

```
SELECT Università.Regione,ROUND(CAST(avg(data-data_positività)as integer)) as
Media_giorni_positività
FROM Positivo JOIN Tampone
ON Positivo.id_caso = Tampone.caso_riferimento
JOIN Persona
ON Positivo.Persona = Persona.Codice_fiscale
JOIN Università
ON Persona.Università = Università.Nome
WHERE Tampone.Risultato = 'Negativo'
GROUP BY Università.Regione;
```

7.2. Query nidificata complessa: Ricerca vaccinati con due dosi positivi per più di 21 giorni

La query seleziona i dati universitari di studenti e professori che sono risultati positivi al covid per un periodo maggiore ai 21 giorni e hanno ricevuto almeno due dosi di vaccino.

```
SELECT * FROM Persona
WHERE 21<(SELECT data-data_positività FROM Positivo JOIN Tampone
          ON Positivo.id_caso = Tampone.caso_riferimento
          WHERE Tampone.Risultato = 'Negativo' and Positivo.persona =
Persona.Codice_fiscale AND 1<(SELECT count(*) FROM Attestato_Positivo
                                                                    WHERE
Attestato_Positivo.id_caso = Positivo.id_caso)
);
```

7.3. Query insiemistica: Lista aule e stato al momento della ricerca di una particolare università

La query seleziona le aule e lo stato di queste di una particolare università, nel nostro caso Università degli studi di Salerno, al momento della ricerca.

```
SELECT codice_aula,Aula.nome,numero_posti,edificio,'Occupata' as Stato_attuale FROM
Aula JOIN Calendario ON Aula.codice_aula = Calendario.aula JOIN Attività ON
Calendario.codice_attività = Attività.attività_calendario JOIN Università ON
Aula.Università = Università.Nome WHERE Università ='Università degli studi di
Salerno' AND  CURRENT_TIME(0) BETWEEN Calendario.orario_inizio AND
Calendario.orario_fine AND Attività.modalità != 'Distanza' AND
Attività.data_svolgimento = CURRENT_DATE
UNION
SELECT codice_aula,Aula.nome,numero_posti,edificio, 'Libera' as Stato_attuale FROM
Aula JOIN Università ON Aula.Università = Università.Nome AND Università =
'Università degli studi di Salerno'
EXCEPT
SELECT codice_aula,Aula.nome,numero_posti,edificio,'Libera' as Stato_attuale FROM
Aula JOIN Calendario ON Aula.codice_aula = Calendario.aula JOIN Attività ON
Calendario.codice_attività = Attività.attività_calendario JOIN Università ON
Aula.Università = Università.Nome WHERE Università ='Università degli studi di
Salerno' AND  CURRENT_TIME(0) BETWEEN Calendario.orario_inizio AND
Calendario.orario_fine AND Attività.modalità != 'Distanza' AND
Attività.data_svolgimento = CURRENT_DATE;
```

Ci è sembrato più leggibile fare una selezione a seconda dell'Università, ma se si volesse avere una lista di tutte le aule di tutte le università e controllare il loro stato attuale:

```
SELECT codice_aula,Aula.nome,numero_posti,edificio,università,'Occupata' as
Stato_attuale FROM Aula JOIN Calendario ON Aula.codice_aula = Calendario.aula JOIN
```

```
Attività ON Calendario.codice_attività = Attività.attività_calendario JOIN
Università ON Aula.Università = Università.Nome  AND  CURRENT_TIME(0) BETWEEN
Calendario.orario_inizio AND Calendario.orario_fine AND Attività.modalità !=
'Distanza' AND Attività.data_svolgimento = CURRENT_DATE
UNION
SELECT codice_aula,Aula.nome,numero_posti,edificio,università, 'Libera' as
Stato_attuale FROM Aula JOIN Università ON Aula.Università = Università.Nome
EXCEPT
SELECT codice_aula,Aula.nome,numero_posti,edificio,università,'Libera' as
Stato_attuale FROM Aula JOIN Calendario ON Aula.codice_aula = Calendario.aula JOIN
Attività ON Calendario.codice_attività = Attività.attività_calendario JOIN
Università ON Aula.Università = Università.Nome AND  CURRENT_TIME(0) BETWEEN
Calendario.orario_inizio AND Calendario.orario_fine AND Attività.modalità !=
'Distanza' AND Attività.data_svolgimento = CURRENT_DATE;
```

7.4. Eventuali Altre query

7.4.1. Ricerca delle attività svolte oggi dai positivi odierni e tracciamento dei contatti

In questa query selezioniamo tutti gli studenti che hanno avuto un contatto oggi, tramite un'attività, con uno studente o con un professore positivo e inoltre tutti i professori che hanno avuto un contatto con uno studente (dato che non si può avere un contatto fra professori dato il vincolo che ogni attività è svolta da un solo professore). Infine, questi contatti vengono inseriti nella tabella Contatto e dovranno caricare il vaccino ed effettuare una quarantena che dipende dal numero di dosi effettuate e da quanto risale l'ultima somministrazione.

```
INSERT INTO Contatto (persona) (SELECT Studente as contatto FROM Prenotazione WHERE
(data_attività,attività) IN (SELECT DISTINCT data_attività,attività
                                FROM Prenotazione
                                WHERE data_attività = CURRENT_DATE AND Studente IN
(SELECT Persona.Codice_fiscale
                                                        FROM Positivo,Persona,Università
WHERE Positivo.persona = Persona.Codice_fiscale AND Persona.Università =
Università.Nome AND Università.Nome = 'Università degli studi di Salerno' AND
Positivo.data_positività = CURRENT_DATE
) UNION (SELECT DISTINCT data_svolgimento as data_attività,attività_calendario as
attività FROM Attività WHERE data_svolgimento = CURRENT_DATE AND professore
IN(SELECT Persona.Codice_fiscale
                                                        FROM Positivo,Persona,Università
WHERE Positivo.persona = Persona.Codice_fiscale AND Persona.Università =
Università.Nome AND Università.Nome = 'Università degli studi di Salerno' AND
Positivo.data_positività = CURRENT_DATE)) )
                                                            AND studente IN (SELECT
Persona.Codice_fiscale
                                                            FROM Persona
```

```
                                                    WHERE stato = 'Libero')) UNION
    (SELECT Professore as contatto FROM Attività WHERE
(data_svolgimento,attività_calendario) IN (SELECT DISTINCT data_attività,attività
                                FROM Prenotazione
                            WHERE data_attività = CURRENT_DATE AND Studente IN
(SELECT Persona.Codice_fiscale
                                                FROM Positivo,Persona,Università
WHERE Positivo.persona = Persona.Codice_fiscale AND Persona.Università =
Università.Nome AND Università.Nome = 'Università degli studi di Salerno' AND
Positivo.data_positività = CURRENT_DATE
)

                                                    AND professore IN (SELECT
Persona.Codice_fiscale
                                                    FROM Persona
                                                    WHERE stato = 'Libero')))
;
```

Abbiamo deciso, anche in questo caso di dividere per Università, inserendo opportunatamente il nome di essa nella clausola where, ma se volessimo tracciare i contatti per tutte le università basterebbe rimuovere la clausola e otterremmo:

```
INSERT INTO Contatto (persona) (SELECT Studente as contatto FROM Prenotazione WHERE
(data_attività,attività) IN (SELECT DISTINCT data_attività,attività
                                FROM Prenotazione
                                WHERE data_attività = CURRENT_DATE AND Studente IN
(SELECT Persona.Codice_fiscale
                                                FROM Positivo,Persona,Università
WHERE Positivo.persona = Persona.Codice_fiscale AND Persona.Università =
Università.Nome  AND Positivo.data_positività = CURRENT_DATE
) UNION (SELECT DISTINCT data_svolgimento as data_attività,attività_calendario as
attività FROM Attività WHERE data_svolgimento = CURRENT_DATE AND professore
IN(SELECT Persona.Codice_fiscale
                                                FROM Positivo,Persona,Università
WHERE Positivo.persona = Persona.Codice_fiscale AND Persona.Università =
Università.Nome  AND Positivo.data_positività = CURRENT_DATE))
)
                                                    AND studente IN (SELECT
Persona.Codice_fiscale
                                                    FROM Persona
                                                    WHERE stato = 'Libero')) UNION
    (SELECT Professore as contatto FROM Attività WHERE
(data_svolgimento,attività_calendario) IN (SELECT DISTINCT data_attività,attività
                                FROM Prenotazione
                            WHERE data_attività = CURRENT_DATE AND Studente IN
(SELECT Persona.Codice_fiscale
```

```
                                                FROM Positivo,Persona,Università
WHERE Positivo.persona = Persona.Codice_fiscale AND Persona.Università =
Università.Nome  AND Positivo.data_positività = CURRENT_DATE
)

                                                    AND professore IN (SELECT
Persona.Codice_fiscale
                                                    FROM Persona
                                                    WHERE stato = 'Libero')))
;
```

8. Viste

Workpackage	***Task***	***Responsabile***
WP4	Viste	Milone Marco

8.1. Vista *StatoUniversità*

La vista in questione ci mostra la lista delle varie università con annesso il numero di iscritti, il numero di positivi al momento, l'indice di positività attuale, inteso come il rapporto fra il numero di positivi e il numero di iscritti, i posti disponibili dell'università e le altre informazioni della stessa.

```
CREATE View VistaSupporto(Università,Numero_iscritti) as (SELECT Università.Nome,
count(Persona.Codice_fiscale) FROM Università,Persona WHERE Università.Nome =
Persona.Università  GROUP BY Università.Nome) ;

CREATE View StatoUniversità(Università,Numero_iscritti,
Numero_positivi,Indice_positività, Posti_disponibili,Regione,Città,Via,Cap,Civico)
as (SELECT
VistaSupporto.università,VistaSupporto.numero_iscritti,count(Persona.Codice_fiscale
),ROUND(CAST(CAST(count(Persona.Codice_fiscale) as float) as
numeric)/numero_iscritti,2),posti_disponibili,regione,città,via,cap,civico FROM
VistaSupporto,Persona,Università WHERE VistaSupporto.università =
Persona.Università AND VistaSupporto.Università = Università.Nome AND Persona.Stato
= 'Positivo' GROUP BY
VistaSupporto.università,VistaSupporto.numero_iscritti,Università.posti_disponibili
,Università.regione,Università.cap,Università.via,Università.civico,Università.citt
à);
```

8.1.1. Query con Vista: Età media degli studenti positivi attualmente nell'università con più casi covid al momento

La query in questiona seleziona gli studenti che sono positivi attualmente, facenti parte dell'università con attualmente più positivi fra tutte e calcola la media della loro età.

```
SELECT Persona.Università as Università_con_più_casi,CAST(avg(CURRENT_DATE-
data_di_nascita)/365 as integer) as Età_media_positivi FROM Persona
WHERE Università IN (SELECT università FROM StatoUniversità
                     WHERE numero_positivi = (SELECT max(numero_positivi) FROM
StatoUniversità))
                     AND Stato = 'Positivo' AND EXISTS (SELECT persona FROM
Studente WHERE studente.persona = persona.codice_fiscale) GROUP BY
Persona.Università;
```

9. Trigger

- Alcuni trigger di inizializzazione hanno alcuni controlli che si riferiscono ad alcuni vincoli aziendali, per completezza abbiamo aggiunto la nota (CHECK AZIENDALE) per specificare quali di questi controlli hanno una natura prettamente derivante da ciò. Viceversa, alcuni trigger aziendali hanno alcuni controlli che si riferiscono all'inizializzazione di alcune occorrenze, in questo caso abbiamo inserimento la nota (CHECK INIZIALIZZAZIONE) per specificare anche qui la natura dei controlli.

9.1. Trigger inizializzazione: *Università Valida*

Workpackage	***Task***	***Responsabile***
WP1	Trigger inizializzazione/popolamento database	Intonti Giovanni

Il trigger in questione controlla, all'inserimento di un'università, se è presente almeno un'aula, uno studente e un professore perché per esistere un'università ha bisogno di almeno un'occorrenza per ognuna di queste entità.

```
CREATE OR REPLACE FUNCTION controllo_università()
RETURNS TRIGGER AS
$BODY$
    DECLARE
        professore integer;
        studente integer;
        aula integer;
    BEGIN
        SELECT count(*) INTO aula FROM Aula WHERE Aula.Università = NEW.Nome;
        SELECT count(*) INTO professore FROM Persona JOIN Professore ON
Persona.Codice_fiscale = Professore.Persona WHERE Persona.Università = NEW.Nome;
        SELECT count(*) INTO studente FROM Persona JOIN Studente ON
Persona.Codice_fiscale = Studente.Persona WHERE Persona.Università = NEW.Nome;
        IF ( aula = 0 or professore = 0 or studente =0 ) THEN RAISE EXCEPTION
$$'Non è possibile inserire % poichè non ha almeno un professore,un aula e uno
studente'$$, NEW.Nome;
        ELSE return NEW;
        END IF;

    END;
$BODY$
LANGUAGE PLPGSQL;

CREATE CONSTRAINT TRIGGER università_valida
AFTER INSERT
```

```
ON Università
DEFERRABLE INITIALLY DEFERRED
FOR EACH ROW
EXECUTE PROCEDURE controllo_università();
```

9.1.2 Trigger inizializzazione: Persona_valida

Workpackage	***Task***	***Responsabile***
WP1	Trigger inizializzazione/popolamento database	Intonti Giovanni

Questo trigger verifica, all'inserimento di una persona nell'università, che essa sia uno studente o un professore

```
CREATE OR REPLACE FUNCTION controllo_persona()
RETURNS TRIGGER AS
$BODY$
    DECLARE
        professore integer;
        studente integer;
    BEGIN
        SELECT count(*) INTO professore FROM Persona JOIN Professore ON
Persona.Codice_fiscale = Professore.Persona WHERE Professore.Persona =
NEW.Codice_fiscale;
        SELECT count(*) INTO studente FROM Persona JOIN Studente ON
Persona.Codice_fiscale = Studente.Persona WHERE Studente.Persona =
NEW.Codice_fiscale;
        IF (professore = 0 and studente =0 ) THEN RAISE EXCEPTION $$'Non è
possibile inserire % poichè non è uno studente o un professore'$$,
NEW.Codice_fiscale;
        ELSE return NEW;
        END IF;

    END;
$BODY$
LANGUAGE PLPGSQL;

CREATE CONSTRAINT TRIGGER persona_valida
AFTER INSERT
ON Persona
DEFERRABLE INITIALLY DEFERRED
FOR EACH ROW
EXECUTE PROCEDURE controllo_persona();
```

9.1.3 Trigger inizializzazione: Professore_no_studente e studente_no_professore

Workpackage	***Task***	***Responsabile***
WP1	Trigger inizializzazione/popolamento database	Intonti Giovanni

Questo trigger controlla, all'inserimento di un nuovo studente o professore, che esso non sia già presente nell'altra tabella; infatti, una persona può essere soltanto uno studente o un professore, non entrambi contemporaneamente.

```
CREATE OR REPLACE FUNCTION no_studente_professore()
RETURNS TRIGGER AS
$BODY$
    DECLARE
        professore integer;
        studente integer;
    BEGIN
        SELECT count(*) INTO professore FROM Professore WHERE Persona =
NEW.Persona;
        SELECT count(*) INTO studente FROM Studente WHERE Persona = NEW.Persona;
        IF ((professore + studente)>1 ) THEN RAISE EXCEPTION $$'Non è possibile
inserire % poichè è già uno studente o un professore'$$, NEW.Persona;
        ELSE return NEW;
        END IF;

    END;
$BODY$
LANGUAGE PLPGSQL;

CREATE CONSTRAINT TRIGGER studente_no_professore
AFTER INSERT
ON Studente
DEFERRABLE INITIALLY DEFERRED
FOR EACH ROW
EXECUTE PROCEDURE no_studente_professore();

CREATE CONSTRAINT TRIGGER professore_no_studente
AFTER INSERT
ON Professore
DEFERRABLE INITIALLY DEFERRED
FOR EACH ROW
EXECUTE PROCEDURE no_studente_professore();
```

9.1.4 Trigger inizializzazione: Elimina_persona e Elimina_aula

Workpackage	***Task***	***Responsabile***
WP1	Trigger inizializzazione/popolamento database	Intonti Giovanni

Nonostante sia di "inizializzazione" serve per preservare le condizioni specificate nell'inizializzazione.
Questo trigger verifica, all'eliminazione di uno studente,un professore o un'aula, se i vincoli per l'università (di averne uno per ogni entità) siano ancora rispettati

```
CREATE OR REPLACE FUNCTION eliminazione_in_università()
RETURNS TRIGGER AS
$BODY$
    DECLARE
        professore integer;
        studente integer;
        aula integer;
        università varchar(50);
    BEGIN
        SELECT Nome INTO università FROM Università WHERE Nome = OLD.Università;
        SELECT count(*) INTO aula FROM Aula WHERE Aula.Università = OLD.Università;
        SELECT count(*) INTO professore FROM Persona JOIN Professore ON
Persona.Codice_fiscale = Professore.Persona WHERE Persona.Università =
OLD.Università;
        SELECT count(*) INTO studente FROM Persona JOIN Studente ON
Persona.Codice_fiscale = Studente.Persona WHERE Persona.Università =
OLD.Università;
        IF ( (aula = 0 or professore = 0 or studente =0) and università is not
null) THEN RAISE EXCEPTION $$'Non è possibile eliminare % poichè altrimenti l
università non ha almeno un professore,un aula e uno studente'$$, OLD;
        ELSE return NEW;
        END IF;

    END;
$BODY$
LANGUAGE PLPGSQL;

CREATE CONSTRAINT TRIGGER elimina_aula
AFTER DELETE OR UPDATE
ON Aula
DEFERRABLE INITIALLY DEFERRED
FOR EACH ROW
EXECUTE PROCEDURE eliminazione_in_università();

CREATE CONSTRAINT TRIGGER elimina_persona
AFTER DELETE OR UPDATE
ON Persona
DEFERRABLE INITIALLY DEFERRED
```

```
FOR EACH ROW
EXECUTE PROCEDURE eliminazione_in_università();
```

9.1.5 Trigger inizializzazione: Positivo_valido

Workpackage	***Task***	***Responsabile***
WP1	Trigger inizializzazione/popolamento database	Intonti Giovanni

Questo trigger controlla, all'inserimento di un nuovo Positivo, che esso abbia un tampone di riferimento della positività e controlla se la data del tampone è antecedente alla data di annunciata positività. Infine, aggiorna lo stato della persona in Positivo.

```
CREATE OR REPLACE FUNCTION controllo_positivo()
RETURNS TRIGGER AS
$BODY$
    DECLARE
        tampone_iniziale integer;
        data_tampone date;
    BEGIN
        SELECT count(*) INTO tampone_iniziale FROM Tampone WHERE caso_riferimento =
NEW.ID_caso AND Risultato = 'Positivo';
        IF(tampone_iniziale=0) THEN RAISE EXCEPTION $$'Non è possibile inserire %
poichè non ha un riferimento al tampone di positività'$$, NEW.ID_caso;
        ELSE
        SELECT data INTO data_tampone FROM Tampone WHERE caso_riferimento =
NEW.ID_caso AND Risultato = 'Positivo';
        IF(NEW.data_positività < data_tampone) THEN RAISE EXCEPTION $$'Non è
possibile inserire % poichè ha un riferimento al tampone di positività con data
postuma alla segnalazione'$$, NEW.ID_caso;
        ELSIF(NEW.ospedalizzato = true and NEW.sintomatico ='false') THEN RAISE
EXCEPTION $$'Non puoi inserire un caso covid ospedalizzato se non è mai stato
sintomatico'$$;
        ELSE
        UPDATE Persona SET Stato = 'Positivo' WHERE Codice_fiscale = NEW.Persona;
        END IF;
        return NEW;
        END IF;

    END;
$BODY$
LANGUAGE PLPGSQL;

CREATE CONSTRAINT TRIGGER positivo_valido
AFTER INSERT
ON Positivo
DEFERRABLE INITIALLY DEFERRED
FOR EACH ROW
EXECUTE PROCEDURE controllo_positivo();
```

9.1.6 Trigger inizializzazione: Elimina_tampone

Workpackage	***Task***	***Responsabile***
WP1	Trigger inizializzazione/popolamento database	Intonti Giovanni

Questo trigger controlla, all'eliminazione o aggiornamento di un tampone, che esista ancora un riferimento(ovvero un ulteriore tampone) per il caso a cui facevano riferimento ,anche in questo caso più che un trigger di inizializzazione, è un trigger per preservare i vincoli definiti in precedenza.

```
CREATE OR REPLACE FUNCTION controllo_rimozione_tampone()
RETURNS TRIGGER AS
$BODY$
    DECLARE
        tampone_positivo integer;
        tampone_contatto integer;
        caso_positivo integer;
        caso_contatto integer;
    BEGIN
        SELECT ID_Caso INTO caso_positivo FROM Positivo WHERE ID_Caso =
OLD.caso_riferimento;
        SELECT ID_Caso INTO caso_contatto FROM Contatto WHERE ID_Caso =
OLD.contatto_riferimento;
        SELECT count(*) INTO tampone_positivo FROM Tampone WHERE caso_riferimento =
OLD.caso_riferimento AND Risultato = OLD.Risultato;
        SELECT count(*) INTO tampone_contatto FROM Tampone WHERE
contatto_riferimento = OLD.contatto_riferimento;

        IF(OLD.caso_riferimento is null and OLD.contatto_riferimento is not null
and caso_contatto is not null and tampone_contatto != 1)  THEN RAISE EXCEPTION
$$'Non è possibile rimuovere % poichè non esiste un altro riferimento per il caso
covid %'$$, OLD.codice_cun,OLD.contatto_riferimento;
        ELSIF(OLD.caso_riferimento is not null and OLD.contatto_riferimento is null
and caso_positivo is not null and tampone_positivo != 1) THEN RAISE EXCEPTION
$$'Non è possibile rimuovere % poichè non esiste un altro riferimento per il caso
covid %'$$, OLD.codice_cun,OLD.caso_riferimento;
        ELSE return OLD;
        END IF;

    END;
$BODY$
LANGUAGE PLPGSQL;

CREATE CONSTRAINT TRIGGER elimina_tampone
AFTER DELETE OR UPDATE
ON Tampone
DEFERRABLE INITIALLY DEFERRED
FOR EACH ROW
EXECUTE PROCEDURE controllo_rimozione_tampone();
```

9.1.7 Trigger inizializzazione: Elimina_padre_professore e Elimina_padre_studente

Workpackage	***Task***	***Responsabile***
WP1	Trigger inizializzazione/popolamento database	Intonti Giovanni

Questo trigger, che scatta all'eliminazione o modifica di professore o studente, verifica che il vincolo iniziale definito per la persona, ovvero che sia un professore o uno studente, e solo uno dei due, venga rispettato. Anche in questo caso è una verifica per i vincoli postuma all'inserimento.

```
CREATE OR REPLACE FUNCTION rimuovi_specializzazione_persona()
RETURNS TRIGGER AS
$BODY$
    DECLARE
        professore integer;
        studente integer;
        persona char(16);
    BEGIN
        SELECT Codice_fiscale into persona FROM Persona WHERE Codice_fiscale =
OLD.Persona;
        SELECT count(*) INTO professore FROM Persona JOIN Professore ON
Persona.Codice_fiscale = Professore.Persona WHERE Professore.Persona = OLD.Persona;
        SELECT count(*) INTO studente FROM Persona JOIN Studente ON
Persona.Codice_fiscale = Studente.Persona WHERE Studente.Persona = OLD.Persona;
        IF(persona is not null and professore=0 and studente = 0)THEN RAISE
EXCEPTION $$'Non è possibile eliminare % poichè altrimenti non è nè un professore
nè uno studente'$$, OLD.Persona;
        ELSE return NEW;
        END IF;

    END;
$BODY$
LANGUAGE PLPGSQL;


CREATE CONSTRAINT TRIGGER elimina_padre_professore
AFTER DELETE OR UPDATE
ON Professore
DEFERRABLE INITIALLY DEFERRED
FOR EACH ROW
EXECUTE PROCEDURE rimuovi_specializzazione_persona();

CREATE CONSTRAINT TRIGGER elimina_padre_studente
AFTER DELETE OR UPDATE
ON Studente
DEFERRABLE INITIALLY DEFERRED
FOR EACH ROW
EXECUTE PROCEDURE rimuovi_specializzazione_persona();
```

9.1.8 Trigger inizializzazione: Attività_valida

Workpackage	*Task*	*Responsabile*
WP1	Trigger inizializzazione/popolamento database	Intonti Giovanni

Questi trigger verificano che all'inserimento o aggiornamento di un attività essa rispetti i seguenti vincoli: il giorno della data in cui si svolge l'attività deve coincidere con il giorno della settimana in cui è schedulata quell'attività nel calendario, la capienza reale dell'attività deve coincidere con l'aula in cui è svolta l'attività schedulata nel calendario e che il professore che svolge quell'attività appartenga alla stessa università dell'aula in cui è svolta l'attività. Infine, se il professore è in uno stato di quarantena o positività all'inserimento dell'attività (CHECK AZIENDALE) essa viene svolta in modalità a distanza modificando il rispettivo attributo.

```
CREATE OR REPLACE FUNCTION controllo_attività()
RETURNS TRIGGER AS
$BODY$
    DECLARE
        giorno_previsto varchar(9);
        giorno_effettivo varchar(9);
        stato varchar(10);
    BEGIN
        SELECT giorno_settimana INTO giorno_previsto FROM Calendario WHERE
codice_attività = NEW.attività_calendario;
        SELECT  to_char(NEW.data_svolgimento,'Day') INTO giorno_effettivo;
        SELECT Persona.Stato INTO stato FROM Persona WHERE Codice_fiscale =
NEW.Professore;
        IF((giorno_previsto = 'Lunedì' and giorno_effettivo NOT LIKE
'Monday%')   or (giorno_previsto = 'Martedì' and giorno_effettivo NOT LIKE
'Tuesday%')    or (giorno_previsto = 'Mercoledì' and giorno_effettivo NOT LIKE
'Wednesday%')    or(giorno_previsto = 'Giovedì' and giorno_effettivo NOT LIKE
'Thursday%')    or(giorno_previsto = 'Venerdì' and giorno_effettivo NOT LIKE
'Friday%')        )
        THEN RAISE EXCEPTION $$'Non è possibile inserire l ''attività % in data %
poichè non schedulata nel calendario in questo giorno della settimana'$$,
NEW.attività_calendario, NEW.data_svolgimento;
        ELSIF(stato != 'Libero' and NEW.modalità != 'Distanza') THEN NEW.modalità =
'Distanza';

        END IF;
         return NEW;
    END;
$BODY$
LANGUAGE PLPGSQL;


CREATE TRIGGER attività_valida
BEFORE INSERT OR UPDATE
ON Attività
```

```
FOR EACH ROW
EXECUTE PROCEDURE controllo_attività();



CREATE OR REPLACE FUNCTION controllo_attività2()
RETURNS TRIGGER AS
$BODY$
    DECLARE
        capienza integer;
        università_attività varchar(50);
        università_professore varchar(50);
    BEGIN
        SELECT Aula.Università INTO università_attività FROM
Attività,Calendario,Aula WHERE Attività.attività_calendario =
Calendario.Codice_attività AND Calendario.aula = Aula.Codice_aula  AND
Attività.attività_calendario = NEW.attività_calendario;
        SELECT Persona.Università INTO università_professore FROM
Professore,Persona WHERE Professore.Persona = Persona.Codice_fiscale AND
Professore.Persona = NEW.professore;
        SELECT Aula.numero_posti  INTO capienza FROM Calendario JOIN Attività ON
Calendario.codice_attività = Attività.attività_calendario JOIN Aula ON
Calendario.Aula = Aula.Codice_aula WHERE Attività.attività_calendario =
NEW.attività_calendario;
        IF(capienza != NEW.capienza_reale) THEN RAISE EXCEPTION $$'Non è possibile
inserire l ''attività % poichè ha un numero di posti non coincidente con l aula in
cui è schedulata'$$, NEW.attività_calendario;
        ELSIF(università_attività != università_professore) THEN RAISE EXCEPTION
$$'Non è possibile inserire la prenotazione poichè si riferisce ad un attività di
un altra università'$$;

        ELSE
        return NEW;
        END IF;
    END;
$BODY$
LANGUAGE PLPGSQL;



CREATE TRIGGER attività_valida2
AFTER INSERT OR UPDATE
ON Attività
FOR EACH ROW
EXECUTE PROCEDURE controllo_attività2();
```

9.1.9 Trigger inizializzazione: Studente_anno

Workpackage	***Task***	***Responsabile***
WP1	Trigger inizializzazione/popolamento database	Intonti Giovanni

Questo trigger verifica, all'inserimento o aggiornamento di uno studente, che esso si sia iscritto almeno a 17 anni e che la data non sia postuma all'anno corrente, inoltre verifica che l'anno di corso sia valido, a seconda della data di iscrizione e della natura stessa dell'attributo (tra 1 e 5 anno).

```
CREATE OR REPLACE FUNCTION controllo_anno_studente()
RETURNS TRIGGER AS
$BODY$
    DECLARE
        data_nascita date;
        anno_nascita integer;
    BEGIN
        SELECT Data_di_nascita INTO data_nascita FROM Studente JOIN Persona ON
Studente.Persona = Persona.Codice_fiscale WHERE Persona = NEW.Persona;
        SELECT (date_part('year',data_nascita)) INTO anno_nascita;
        IF((NEW.Anno_iscrizione - anno_nascita < 17) or  NEW.Anno_iscrizione >
(date_part('year',now())))
        THEN RAISE EXCEPTION $$'Non è possibile inserire lo studente % poichè è
iscritto in una data non valida'$$, NEW.Persona;
        ELSIF((NEW.Anno_corso >  ((date_part('year',now()) -
NEW.Anno_iscrizione)+1)) or (NEW.Anno_corso <0 or NEW.Anno_corso>5))
        THEN RAISE EXCEPTION $$'Non è possibile inserire lo studente % poichè è
iscritto ad un anno non valido per lui'$$, NEW.Persona;
        ELSE return NEW;
        END IF;

    END;
$BODY$
LANGUAGE PLPGSQL;


CREATE CONSTRAINT TRIGGER studente_anno
AFTER INSERT OR UPDATE
ON Studente
DEFERRABLE INITIALLY DEFERRED
FOR EACH ROW
EXECUTE PROCEDURE controllo_anno_studente();
```

9.1.10 Trigger inizializzazione: Prenotazione_valida

Workpackage	***Task***	***Responsabile***
WP1	Trigger inizializzazione/popolamento database	Intonti Giovanni

Questo trigger verifica, all'inserimento o modifica di una prenotazione, i seguenti vincoli: l'attività che si sta cercando di prenotare non deve essere già trascorsa in termini di data, inoltre per prenotarsi lo studente deve appartenere alla stessa università in cui è svolta l'attività. Infine (CHECK AZIENDALE), se lo studente è già in uno stato di quarantena o positività o il numero di prenotazione supera la capienza predisposta, l'attività è svolta in modalità mista.

```
CREATE OR REPLACE FUNCTION controllo_prenotazione()
RETURNS TRIGGER AS
$BODY$
    DECLARE
        data_attività date;
        università_attività varchar(50);
        università_studente varchar(50);
        stato varchar(10);
        numero_prenotazioni integer;
    BEGIN
        SELECT count(*) INTO numero_prenotazioni FROM Prenotazione WHERE
Prenotazione.data_attività = NEW.data_attività AND attività = NEW.attività;
        SELECT Aula.Università INTO università_attività FROM
Prenotazione,Attività,Calendario,Aula WHERE Prenotazione.data_attività =
Attività.data_svolgimento AND Prenotazione.attività = Attività.attività_calendario
AND Attività.attività_calendario = Calendario.Codice_attività AND Calendario.aula =
Aula.Codice_aula  AND Prenotazione.data_attività = NEW.data_attività AND
Prenotazione.attività = NEW.attività;
        SELECT Persona.Università INTO università_studente FROM
Prenotazione,Studente,Persona WHERE Prenotazione.studente = Studente.Persona AND
Studente.Persona = Persona.Codice_fiscale AND Prenotazione.studente = NEW.studente;
        SELECT Persona.stato INTO stato FROM Persona WHERE Codice_fiscale =
NEW.studente;
        IF(NEW.data_attività < CURRENT_DATE)
        THEN RAISE EXCEPTION $$'Non è possibile inserire la prenotazione poichè si
riferisce ad un attività già trascorsa'$$;
        ELSIF(università_attività != università_studente) THEN RAISE EXCEPTION
$$'Non è possibile inserire la prenotazione poichè si riferisce ad un attività di
un altra università'$$;
        ELSIF(stato = 'Quarantena' or stato ='Positivo' or numero_prenotazioni >
(SELECT capienza_predisposta FROM Attività WHERE attività_calendario = NEW.attività
AND data_svolgimento = NEW.data_attività ))  THEN
        UPDATE Attività SET modalità = 'Mista' WHERE data_svolgimento =
NEW.data_attività AND attività_calendario = NEW.attività;

        END IF;
        return NEW;

    END;
$BODY$
LANGUAGE PLPGSQL;
```

```
CREATE CONSTRAINT TRIGGER prenotazione_valida
AFTER INSERT OR UPDATE
ON Prenotazione
DEFERRABLE INITIALLY DEFERRED
FOR EACH ROW
EXECUTE PROCEDURE controllo_prenotazione();
```

9.2. Trigger per vincoli aziendali

Workpackage	***Task***	***Responsabile***
WP4	Trigger per vincoli aziendali	Milone Marco

La sezione deve contenere almeno due trigger. È possibile aggiungerne altri se necessario

9.2.1. Trigger1: Tampone_valido

Il trigger in questione verifica, all'inserimento o aggiornamento di un tampone, che, nel caso in cui esso si riferisca ad un contatto, esso non arrivi dopo un tempo inferiore al periodo previsto di quarantena per un contatto, se invece il periodo è valido e il tampone è negativo, modifichiamo lo stato in Libero, se è positivo, lo inseriamo tra i positivi e modifichiamo il suo stato in Positivo. Se il tampone inserito, invece, fa riferimento ad un caso covid positivo ed è negativo, allora modifichiamo lo stato in Libero.
(CHECK INIZIALIZZAZIONE) Nel caso il tampone si riferisca ad un caso di positività la data del tampone negativo non deve essere antecedente a quella del tampone positivo. Inoltre un tampone con riferimenti sia ad un caso positivo che ad un contatto oppure a nessuno dei due non viene accettato.

```
CREATE OR REPLACE FUNCTION controllo_tampone()
RETURNS TRIGGER AS
$BODY$
    DECLARE
        tampone_positivo integer;
        tampone_negativo integer;
        tampone_contatto integer;
        data_somministrazione date;
        numero_vaccini integer;
        data_contatto date;
        data_positività date;
        contatore integer;
        nuovo_tampone integer;
    BEGIN
        SELECT max(id_caso) into nuovo_tampone FROM Positivo;
        IF((SELECT count(*) FROM Positivo) = 0) THEN contatore = 0;
```

```
        ELSE contatore = 1;
        END IF;
        SELECT Data INTO data_positività FROM Tampone WHERE risultato = 'Positivo'
AND caso_riferimento = NEW.caso_riferimento;
        SELECT Attestato_contatto.data_somministrazione INTO data_somministrazione
FROM Attestato_contatto WHERE ID_caso = NEW.contatto_riferimento ORDER BY
Attestato_contatto.data_somministrazione DESC LIMIT 1;
        SELECT count(*) INTO numero_vaccini FROM Attestato_contatto WHERE
Attestato_contatto.ID_caso = NEW.contatto_riferimento;
        SELECT Contatto.data_contatto INTO data_contatto FROM Contatto WHERE
ID_caso = NEW.contatto_riferimento;
        IF(NEW.caso_riferimento is null AND NEW.contatto_riferimento is null)
            THEN RAISE EXCEPTION $$'Non è possibile inserire % poichè non ha un
riferimento a nessun caso covid'$$, NEW.codice_cun;
        ELSIF(NEW.caso_riferimento is not null AND NEW.contatto_riferimento is not
null)
        THEN RAISE EXCEPTION $$'Non è possibile inserire % poichè non si può avere
un riferimento ad un positivo ed ad un contatto'$$, NEW.codice_cun;
        ELSIF(NEW.contatto_riferimento is not null AND numero_vaccini >= 3 AND
(NEW.Data-data_contatto)<7) THEN RAISE EXCEPTION $$'Non è possibile inserire %
poichè per il caso % è prevista una quarantena maggiore'$$,
NEW.codice_cun,NEW.contatto_riferimento;
        ELSIF(NEW.contatto_riferimento is not null AND numero_vaccini = 2 AND
(NEW.Data-data_contatto<7) AND (CURRENT_DATE-data_somministrazione < 120)) THEN
RAISE EXCEPTION $$'Non è possibile inserire % poichè per il caso % è prevista una
quarantena maggiore'$$, NEW.codice_cun,NEW.contatto_riferimento;
        ELSIF(NEW.contatto_riferimento is not null AND numero_vaccini = 2 AND
(NEW.Data-data_contatto<10) AND (CURRENT_DATE-data_somministrazione > 120)) THEN
RAISE EXCEPTION $$'Non è possibile inserire % poichè per il caso % è prevista una
quarantena maggiore'$$, NEW.codice_cun,NEW.contatto_riferimento;
        ELSIF(NEW.contatto_riferimento is not null AND numero_vaccini < 2 AND
(NEW.Data-data_contatto)<10) THEN RAISE EXCEPTION $$'Non è possibile inserire %
poichè per il caso % è prevista una quarantena maggiore'$$,
NEW.codice_cun,NEW.contatto_riferimento;
        ELSIF(NEW.Risultato = 'Negativo' and NEW.Data < data_positività) THEN RAISE
EXCEPTION $$'Non è possibile inserire % poichè un tampone di accertata negatività
non può avere una data antecedente alla positività'$$, NEW.codice_cun;
        ELSIF(NEW.Risultato = 'Negativo' and NEW.contatto_riferimento is not null)
THEN
            UPDATE Persona SET Stato = 'Libero' WHERE Codice_fiscale = (SELECT
DISTINCT persona FROM Contatto, Tampone WHERE id_caso = NEW.contatto_riferimento);
        ELSIF(NEW.Risultato = 'Negativo' and NEW.caso_riferimento is not null) THEN
            UPDATE Persona SET Stato = 'Libero' WHERE Codice_fiscale = (SELECT
DISTINCT persona FROM Positivo, Tampone WHERE id_caso = NEW.caso_riferimento);
        ELSIF(NEW.Risultato = 'Positivo' and NEW.contatto_riferimento is not null
AND contatore = 1) THEN
            UPDATE Persona SET Stato = 'Positivo' WHERE Codice_fiscale IN (SELECT
DISTINCT persona FROM Contatto, Tampone WHERE id_caso = NEW.contatto_riferimento);
```

```
            INSERT INTO Positivo
VALUES(nuovo_tampone+1,DEFAULT,DEFAULT,NEW.Data,(SELECT DISTINCT persona FROM
Contatto, Tampone WHERE id_caso = NEW.contatto_riferimento));
            INSERT INTO Tampone
VALUES(DEFAULT,NEW.Data,NEW.Orario,'Positivo',NEW.tipologia,nuovo_tampone+1,NULL);
        ELSIF(NEW.Risultato = 'Positivo' and NEW.contatto_riferimento is not null
AND contatore = 0) THEN
            UPDATE Persona SET Stato = 'Positivo' WHERE Codice_fiscale IN (SELECT
DISTINCT persona FROM Contatto, Tampone WHERE id_caso = NEW.contatto_riferimento);
            INSERT INTO Positivo VALUES(1,DEFAULT,DEFAULT,NEW.Data,(SELECT DISTINCT
persona FROM Contatto, Tampone WHERE id_caso = NEW.contatto_riferimento));
            INSERT INTO Tampone
VALUES(DEFAULT,NEW.Data,NEW.Orario,'Positivo',NEW.tipologia,1,NULL);

        END IF;
        return NEW;

    END;
$BODY$
LANGUAGE PLPGSQL;

CREATE CONSTRAINT TRIGGER tampone_valido
AFTER INSERT OR UPDATE
ON Tampone
DEFERRABLE INITIALLY DEFERRED
FOR EACH ROW
EXECUTE PROCEDURE controllo_tampone();
```

9.2.2 Trigger 2: Tampone_Duplicato

Questo trigger verifica che per ogni positivo ci sia soltanto un tampone relativo alla positività ed uno relativo alla negatività, parallelamente per un contatto verifica l'esistenza di un solo tampone (se è negativo il caso è concluso, se è positivo è concluso e inizia un caso di positività)

```
CREATE OR REPLACE FUNCTION controllo_tampone_duplicato()
RETURNS TRIGGER AS
$BODY$
    DECLARE
        tampone_positivo integer;
        tampone_negativo integer;
        tampone_contatto integer;
    BEGIN

        IF(NEW.contatto_riferimento is not null) THEN SELECT count(*) INTO
tampone_contatto FROM Tampone WHERE contatto_riferimento =
NEW.contatto_riferimento;    END IF;
        IF(NEW.caso_riferimento is not null) THEN SELECT count(*) INTO
tampone_positivo FROM Tampone WHERE caso_riferimento = NEW.caso_riferimento AND
Risultato = 'Positivo';    END IF;
```

```
        IF(NEW.caso_riferimento is not null) THEN SELECT count(*) INTO
tampone_negativo FROM Tampone WHERE caso_riferimento = NEW.caso_riferimento AND
Risultato = 'Negativo';   END IF;
        IF(NEW.contatto_riferimento is not null AND tampone_contatto > 1) THEN
RAISE EXCEPTION $$'Non è possibile inserire % poichè esiste già un tampone con
riferimento lo stesso caso covid'$$, NEW.codice_cun;   END IF;
        IF(NEW.caso_riferimento is not null AND (tampone_positivo > 1 OR
tampone_negativo >1)) THEN  RAISE EXCEPTION $$'Non è possibile inserire % poichè
esiste già un tampone con riferimento lo stesso caso covid'$$,
NEW.codice_cun;   END IF;
        return NEW;


    END;
$BODY$
LANGUAGE PLPGSQL;


CREATE CONSTRAINT TRIGGER tampone_duplicato
AFTER INSERT OR UPDATE
ON Tampone
DEFERRABLE INITIALLY DEFERRED
FOR EACH ROW
EXECUTE PROCEDURE controllo_tampone_duplicato();
```

9.2.3 Trigger 3: Cambio_stato

Questo trigger verifica, all'aggiornamento di un positivo se lo stato è cambiato, in quel caso nel caso è uno studente, cerchiamo le attività future a cui è prenotato che sono ancora svolte in presenza e le rendiamo svolte in modalità mista, altrimenti nel caso in cui è un professore, selezioniamo le attività future svolte da lui e modifichiamo la modalità rendendola a distanza.

```
CREATE OR REPLACE FUNCTION cambio_attività()
RETURNS TRIGGER AS
$BODY$
    DECLARE
        professore integer;
        data_partenza date;
    BEGIN
        SELECT count(*) INTO professore FROM Professore WHERE Persona =
NEW.Codice_fiscale;
        IF(NEW.stato != OLD.stato) THEN
            IF(NEW.stato = 'Quarantena') THEN SELECT data_contatto INTO
data_partenza FROM Contatto WHERE persona = NEW.Codice_fiscale ORDER BY
data_contatto DESC LIMIT 1;
```

```
            ELSIF(NEW.stato = 'Positivo') THEN SELECT data_positività INTO
data_partenza FROM Positivo WHERE persona = NEW.Codice_fiscale ORDER BY
data_positività DESC LIMIT 1;
            END IF;

            IF(professore = 0 AND (NEW.stato ='Quarantena' or NEW.stato
='Positivo')) THEN
                UPDATE Attività SET Modalità = 'Mista' WHERE
(data_svolgimento,attività_calendario) IN (select data_attività,attività FROM
Prenotazione,Attività WHERE data_attività = data_svolgimento AND attività =
attività_calendario AND studente = NEW.Codice_fiscale AND modalità ='Presenza' AND
data_svolgimento >= data_partenza );

            ELSIF(professore=1 AND (NEW.stato='Quarantena' or
NEW.stato='Positivo')) THEN
                UPDATE Attività SET Modalità = 'Distanza' WHERE Attività.professore
= NEW.Codice_fiscale AND data_svolgimento >= data_partenza;

            END IF;
        END IF;
        return NEW;

    END;
$BODY$
LANGUAGE PLPGSQL;

CREATE CONSTRAINT TRIGGER cambio_stato
AFTER UPDATE
ON Persona
DEFERRABLE INITIALLY DEFERRED
FOR EACH ROW
EXECUTE PROCEDURE cambio_attività();
```

9.2.4 Trigger 4: Contatto_validazione

Trigger che modifica lo stato delle persone risultate contatti di positivi

```
CREATE OR REPLACE FUNCTION controllo_contatto()
RETURNS TRIGGER AS
$BODY$
    DECLARE

    BEGIN
        UPDATE Persona SET Stato = 'Quarantena' WHERE Codice_fiscale = NEW.Persona;

        return NEW;

    END;
```

```
$BODY$
LANGUAGE PLPGSQL;


CREATE CONSTRAINT TRIGGER contatto_validazione
AFTER INSERT
ON Contatto
DEFERRABLE INITIALLY DEFERRED
FOR EACH ROW
EXECUTE PROCEDURE controllo_contatto();
```

9.2.5 Trigger 5: Reset_positivo e Reset_contatto

Entrambi questi trigger ristabiliscono lo stato naturale per quelle persone che vengono eliminate da una delle due tabelle senza avere un altro caso attivo al momento.

```
CREATE OR REPLACE FUNCTION cancellazione_positivo()
RETURNS TRIGGER AS
$BODY$
    DECLARE
        check_positivo integer;
        check_negativo integer;
    BEGIN
        SELECT count(*) INTO check_positivo FROM Positivo JOIN Tampone ON
Positivo.id_caso = Tampone.caso_riferimento WHERE id_caso = OLD.id_caso AND
Tampone.Risultato = 'Positivo';
        SELECT count(*) INTO check_negativo FROM Positivo JOIN Tampone ON
Positivo.id_caso = Tampone.caso_riferimento WHERE id_caso = OLD.id_caso AND
Tampone.Risultato = 'Negativo';
        IF(check_positivo = check_negativo) THEN UPDATE Persona SET Stato =
'Libero' WHERE Stato = 'Positivo' AND Codice_fiscale = OLD.persona;


        END IF;
        RETURN NEW;
    END;
$BODY$
LANGUAGE PLPGSQL;


CREATE CONSTRAINT TRIGGER reset_positivo
```

```
AFTER UPDATE OR DELETE
ON Positivo
DEFERRABLE INITIALLY DEFERRED
FOR EACH ROW
EXECUTE PROCEDURE cancellazione_positivo();



CREATE OR REPLACE FUNCTION cancellazione_contatto()
RETURNS TRIGGER AS
$BODY$
    DECLARE
        altro_contatto integer;
    BEGIN
        SELECT count(*) INTO altro_contatto FROM Contatto LEFT JOIN Tampone ON
Contatto.id_caso = Tampone.contatto_riferimento WHERE persona = OLD.persona AND
Tampone.Risultato is null;
        IF(altro_contatto=0) THEN UPDATE Persona SET Stato = 'Libero' WHERE Stato =
'Quarantena' AND Codice_fiscale = OLD.persona;


        END IF;
        RETURN NEW;
    END;
$BODY$
LANGUAGE PLPGSQL;



CREATE CONSTRAINT TRIGGER reset_contatto
AFTER UPDATE OR DELETE
ON Contatto
DEFERRABLE INITIALLY DEFERRED
FOR EACH ROW
EXECUTE PROCEDURE cancellazione_contatto();
```