|  |
| --- |
| **Appunti**  **CORSO RESKILLING JAVA - APPUNTI** |
|  |
| **Redatto da:** Marco Cocciarini  **Verificato da:**  **Approvato da:**  **Data di validità:** 15/09/2025  **N.ro versione:** 1.0  **N.ro pagine:** 96  **Distribuzione:** Sito Intranet Aziendale  Nome File: CorsoReskillingJAVA\_Appunti.docx |
|  |

|  |
| --- |
| **Attenzione**  Il presente documento è disponibile in copia originale sul server della rete aziendale.  Ogni copia cartacea si ritiene copia di lavoro **non controllata**.  E’ responsabilità di chi utilizza copie non controllate verificarne il livello di aggiornamento. |

Sommario

[Indice Delle Figure 3](#_Toc215047074)

[Non è stata trovata alcuna voce dell'indice delle figure. 4](#_Toc215047075)

[1 AGGIORNAMENTI DELLA VERSIONE 5](#_Toc215047076)

[2 GLOSSARIO 6](#_Toc215047077)

[3 APPUNTI CORSO 7](#_Toc215047078)

[3.1 FRAMEWORK JBF : 07/11/2025 7](#_Toc215047079)

[3.1.1 VIDEOLEZIONE 0 : 00 - AMC\_AP\_01 09-13 7](#_Toc215047080)

[3.1.2 VIDEOLEZIONE 1 : 01 - AREAS\_AP\_01\_JBF 09-13 40](#_Toc215047081)

[3.1.3 VIDEOLEZIONE 2 : 02 - AREAS\_AP\_01\_JBF 04-18 62](#_Toc215047082)

[3.1.4 VIDEOLEZIONE 3 : 03 - AREAS\_AP\_01\_JBF 09-14 70](#_Toc215047083)

[3.1.5 VIDEOLEZIONE 4 : 04 - AREAS\_AP\_01\_JBF 14-18 88](#_Toc215047084)

[3.1.6 VIDEOLEZIONE 5 : 05 - AREAS\_AP\_01\_JBF 09-13 112](#_Toc215047085)

[3.1.7 VIDEOLEZIONE 6 : 06 - AREAS\_AP\_01\_JBF 14-18 144](#_Toc215047086)

[3.1.8 VIDEOLEZIONE 7 : 07 - AREAS\_AP\_01\_JBF 09-13(1) 163](#_Toc215047087)

[3.1.9 VIDEOLEZIONE 8 : 08 - AREAS\_AP\_01\_JBF 14-18 188](#_Toc215047088)

[3.1.10 VIDEOLEZIONE 9 : 09 - AREAS\_AP\_01\_JBF 09-13 218](#_Toc215047089)

[3.1.11 VIDEOLEZIONE 10 : 10 - AREAS\_AP\_01\_JBF 14-18 232](#_Toc215047090)

[3.1.12 VIDEOLEZIONE 11 : 11 - AREAS\_AP\_01\_JBF 09-13 280](#_Toc215047091)

# Indice Delle Figure

# Non è stata trovata alcuna voce dell'indice delle figure.

# AGGIORNAMENTI DELLA VERSIONE

|  |  |  |  |
| --- | --- | --- | --- |
| **Versione** | **Data** | **Motivo** | **Modifiche** |
| **1.0** | 15/09/2025 | Prima emissione del documento | n.a. |

# GLOSSARIO

|  |  |
| --- | --- |
| **Sigla** | **Descrizione** |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

# APPUNTI CORSO

## FRAMEWORK JBF : 07/11/2025

Prodotti nostri si appoggiano su un framework proprietario

Questa parte la facciamo internamente : basato su struts

Framework creato partendo da quelle basi. Framework JBF

Questo corso non sara con classroom : videocorsi da fare in self study.

Poi ci sarà una persona di riferimento a cui possiamo chiedere le cose che non abbiamo capito o approfondimenti del caso

Daniele dall0’osso : AREAS parte clinica ospedalier a e territoriale. Protno soccorso, cartella clinica ricoveri etc + territoriale per medicina di base e assistenza territoriale.

Paola giovannetti : prenotazione di prestazioni sanitarie (CUP e portali di prenotazione per il cittadino) – CUP regionali 🡺 problema delle performance. Integrazione con pagopa per il pagamento etc…integrazioni con i totem, con i sistemi SMS per mandare sms a cittadini, app IO.

2 sistemi di prenotazione : uno da dismettere (è del 1999) , e uno su AREAS in produzione dal 2010 : molti sviluppi in corso su questo, e ultimamente di questa versione areas è stato fatto il frontend nuovo su ellipse (nuova piattaforma). Per ora il backend è su AREAS :il frontend ellipse tramite servizi si interfaccia con il backend AREAS.

Ellipse è a microservizi, AREAS no.

AMC è in produzione dal 2007.

Valerio d’angiolella : nostra interfaccia con cui dialogare e organizzerà sessioni di un’oretta al giorno per sentirci con lui per rispondere alle nostre domande.

Valerio lavora con daniele dall’osso : verremo abilitati aun canale con le videolezioni da 4 ore ciascuno : una decina di video in totale. Dovrebbero occupare in totale una settimana. + 30%

Possiamo anche usare la chat del canale eventualmente per le domande.

Dopo decideranno come collocarci sui vari gruppi.

Cosa fondamentale: nel giorno 1 la configurazione dell’ambiente di sviluppo in locale. Vedere a fine giornata chi è riuscito a configurarlo e farlo funzionare e chi no, sistemando l’ambiente di sviluppo locale.

### 

### VIDEOLEZIONE 9 : 09 - AREAS\_AP\_01\_JBF 09-13

Parte iniziale : verifica configurazione eclipse locale.

Rifare i primi esercizi MVC 🡪 ripetuti su un’ottica diversa. Perché in realtà ci sono ogni tanto delle maschere fatte in maniera diversa : es 🡪 logistica -🡪 stampe 🡪 nuova gestione movimenti ricevimenti

Immagine che contiene testo, software, numero, Icona del computer

Il contenuto generato dall'IA potrebbe non essere corretto.

Queste maschere vengono chiamate MASCHERE JAUF: perché a un certo punto del tempo è stata introdotta una nuova libreria grafica detta JAUF. Questo è il lookandfeel delle maschere nuove.

Ripeteremo tutti gli esercizi ma nel nuovo formato JAUF. La libreria JAUF sta sempre dentro PSGLib: introduce solo novità di tipo interfaccia utente.

Per fare questo tipo di form non si usano nemmeno le pagine JSP.

Per fare le build + veloci: in application.properties ci sono due righe per ogni applicazione 🡪 commentando a coppie (ctrl+shift+7) 🡪 commento le app che voglio skippare dalla build process

Senno in build.properties la riga con numProjects la si valorizza c on il numero di progetti da compilare e sotto c’è la lista di n righe di tipo sourceProjectN

Immagine che contiene testo, software, Pagina Web, Sito Web

Il contenuto generato dall'IA potrebbe non essere corretto.

[**verifiche e correzioni sui vari eclipse FINO A 01:01:30**]

XMPI : che cos’è? Finora di JBF abbiamo visto: PSGLib e JBFReporting, con il progetto Sianc [applicazione da FE = ANAGRAFICHE] , è sinonimo anche di XMPI, sianc, anagrafiche.

Ora vedremo XMPI sia da punto di vista funzionale che vedendo un po’ di tabelle principali in DB e bean più importanti.

**Argomenti** :

* Anagrafiche principali
* Gestione proposte
* Gestione pesi
* Gestione Alias.

Proposte, pesi e alias : sono specifici dell’anagrafica del paziente.

XMPI = è un MPI , ovvero Master Patient Index = soluzione SW per gestione e governo completo dell’anagrafica del paziente. In quanto tale, XMPI non si limita a raccolta dati anagrafici, ma assicura anche la loro consistenza e controllo continuo con funzionalità specifiche .

* Accorpamento anagrafico : individuare in db due posizioni anagrafiche afferenti a stessa persona fisica (duplicati)
* Gestione proposte di variazione : per proteggere il golden record del paziente e le modifiche su questo avvengano quindi in maniera controllata
* Gestione sulla qualifica del dato : gestire chi ha valorizzato un dato e che livello di affidabilità ha quella persona
* Profondità storica sulle variazioni : tenere traccia di tutto ciò che avviene in termini di modifiche all’anagrafica del paziente.

Concetto fondamentale : identificare e condividere anagrafiche all’interno della intera organizzazione tramite appositi servizi 🡺 (integrazione) web services.

XMPI non contiene solo l’anagrafica del paziente, ma può consentire di gestire tutto il patrimonio anagrafico di una azienda, consentendo di codificare soggetti, strutture prodotti e anagrafiche di base , quali ad esempio :

* Provincie
* Regioni
* Comuni e stati esteri
* Asl e presidi ….

Ogni anagrafica prevede un range di abilitazione/disabilitazione 🡪 consente di gestire obsolescenza del dato : es comune che passa da una regione all’altra, disattivo il vecchio e attivo quello nuovo sotto la nuova regione. Sui periodi di inizio e fine validità poi si innestano una serie di controlli incrociati : es il periodo del comune deve essere coperto dal periodo di validità della regione “padre” a cui il comune afferisce. Es analogo: c’è anche un controllo tra periodo di validità di una regione o stato estero e data di nascita persona 🡪 es: se io sono nato in un comune che poi diventa frazione di comune più grande, io rimango essere nato in quel comune, ma i nuovi nati no. Anagrafica comuni : GESTISCE ANCHE GLI STATI ESTERI.

Un dettaglio del comune è l’associazione con la rispettiva ASL , che è anch’essa un’anagrafica.

Le ASL sono associabili a un comune 🡪 poi ci sono aspetti funzionali : es in anagrafica paziente c’è una funzionalità per cui a partire dal comune di residenza del paziente il SW gli assegna la ASL di appartenenza.

Altra funzionalità comune : gestione stradario (BOTTONE)🡪 è per la gestione di comuni grandi che hanno più CAP in base alle zone: posso censire le vie dove c’è un cambio di CAP.

Ciò sul paziente che significa: che se io indico comune di residenza che non ha stradario, nella via di residenza del paziente posso mettere testo libero. Se invece c’è lo stradario attivo, la via di residenza del paziente è selezionabile tramite lookup che insiste sullo stradario di quella città.

[**01:24:05**]

**PANORAMICA ANAGRAFICHE DI BASE:**

Anagrafica Cittadinanze: molto semplice

ASL / presidi : censisco asl e presidi ospedalieri 🡪 ha regione di appartenenza. Poi da anagrafica comune, se vado su comune-asl : vedo solo le asl filtrate per regione, che ha senso associare.

Classi esenzioni : esenzioni pazienti

Rapporti : lo vediamo poi nell’esercizio sui soggetti

Poi anagrafiche semplici

Nucleo Familiare : per gestire il fatto che un paziente è in un nucleo familiare composto da N soggetti, che hanno relazione di un certo tipo tra loro.

Ci sono anche voci menu anagrafiche base dedicate alle forze armate.

Queste anagrafiche di base sono poi utilizzate nelle anagrafiche principali.

[**01:35:00**]

Anagrafica del PAZIENTE : detta anche anagrafica assistiti. È la principale di XMPI.

* Info di base
* Info sull’ASL
* Attività professionali
* Medici di base
* Esenzioni legate al paziente

Guardiamo le tabelle e i campi della anagrafica paziente, vedendo il funzionamento delle TABELLE GENERICHE (argomento di JBF).

Menu : ANAGRAFICHE 🡪 PAZIENTI 🡪 GESTIONE

Nuovo Paziente :

* Info anagrafiche di base e comune di residenza
* CONFERMA
* Paziente salvato : tutti i tab tranne consensi, nucleo familiare e medicina di base che sono di dettaglio, in realtà anche se sono tab separati NON SONO DI DETTAGLIO 🡪 sono stati creati come tab separati (altro controller) ma non sono details : è un controller che gestisce sempre le stesse info di testata : il paziente ha circa 250 campi!!!!
  + tab principale (visibile anche in inserimento) è dati anagrafici
    - Gestione codici straniero per soggetti stranieri
    - Blocco comune residenza
    - Ribaltare residenza come domicilio
  + Tab altri dati : sempre info anagrafiche ma dati più di nicchia per tenere la form principale meno affollata sono stati spostati qua.
    - Numeri di telefono vari
  + Tab ASL : dati assistenza sanitaria
    - Triplette per asl : assistenza, appartenenza, provenienza e destinazione
      * Asl , distretto e regione
  + Tab Professionali:
    - Dati anagrafici relativi a titolo di studio e posizione professionale, datore di lavoro etc…
  + Medicina di base: info del medico di base, c’è anche un bottone MEDICI per gestire i dettagli 🡪 il bottone si abilita solo dopo aver valorizzato la ASL di assistenza.
    - Medici : popup per inserire in una griglia di dettaglio i medici : lo vedremo con esercizio per gestione dei soggetti 🡪 gestione scelta e revoca dei medici
  + Tab STORICO : tiene traccia di tutte le variazioni fatte su questa anagrafica. Per ogni campo vedo valore precedente e valore attuale.
  + **La form di gestione (sia paziente che soggetto) NON HA CAMPI OBBLIGATORI : se ne vedo alcuni con sfondo giallo è perché sono stati impostati come tali tramite la profilatura configurabile presso un certo cliente**.
* **FORM DI RICERCA PAZIENTE** :
  + I filtri non sono in like
  + Devo mettere per forza filtri altrimenti non parte la ricerca 🡪 limite messo per questioni di privacy sui dati.

[01:55:30]

Il paziente è utile per esplorare una funzionalità del framework JBF : le opzioni nel campo combo “STATO CIVILE” non sono stabilite dallo sviluppatore lato codice, e non sono nemmeno prese da una tabella anagrafica puntata come fonte dati 🡪 TABELLE GENERICHE

**TABELLE GENERICHE**: servono per gestire tutti quei dati che sono poco o per nulla movimentati, e che quindi avrebbe poco senso gestire e censire su una tabella dedicata. Usati per popolare combo etc… senza dover cablare i valori lato codice. **SONO ANALOGO DEGLI EDW\_LOV in NETASIAL!!!!**

Li trovo in : **MENU 🡪 CONFIGURAZIONE 🡪 TABELLE DI BASE 🡪 TABELLE GENERICHE**

Mi posso creare una mia tabella generica 🡪 nella testata tabella generica posso poi gestirne gli elementi (i dettagli) che sono i valori. Vanno tutte in una stessa tabella di DB che le ospita tutte : due tabelle una per le testate e una per i dettagli.

* TESTATA
* ELEMENTI : ciascuno ha un codice e una descrizione

**Lato DB**:

* TESTATA 🡪 SI\_TTABGEN : ci entro con TG\_CODE = ‘MIOCODICETABGEN’
* DETTAGLI ELEMENTI 🡪 SI\_ETABGEN
  + EG\_TCOD = CODICE DEL PADRE 🡪 entro con questa in where e vedo tutti gli elementi di una certa tabella generica.

Esempio pratico : stato civile del paziente 🡪 presi da una tabella generica SA\_STATO\_CIVILE

Vado da frontend , apro il dettaglio SA\_STATO\_CIVILE 🡪 aggiungo un elemento ad esempio con codice 999 e salvo.

Torno in anagrafica paziente e vedo il nuovo valore nella combo!

Con questo sistema di tabelle generiche posso anche dare modo all’utente stesso di auto-configurarsi i possibili valori per un certo campo.

[**02:04:30**]

**Come capire quali campi sono oggetto di una tabella generica** ?

O tramite menu avanzate 🡪 stampa registro metadati 🡪 cerco il bean del paziente (l’anagrafica che mi interessa) 🡪 confermo 🡺 mi dà la stampa PDF della descrizione del paziente, nella sezione delle proprietà vedo se ci clicco quale è il lookup collegato 🡪 tramite SI\_LOOKUP vedo quale tabella lo alimenta [sarà SI\_ETABGEN]e inoltre da qui vedo i filtri con cui accede a SI\_ETABGEN.

Modo più veloce: apro l’XML DESCRIPTOR associato al bean 🡪 cerco se ci sono nodi “Lookup” figli di property e se hanno nel sottonodo Filters indicato un filtro su EG\_TCOD :

Immagine che contiene testo, schermata, software, Pagina Web

Il contenuto generato dall'IA potrebbe non essere corretto.

Questa sintassi è sconsigliata perché è scolpita nella roccia, meglio scrivere queste cose nella getObjctInfo() del bean. Utile per i bean esistenti vecchi.

Altro modo consigliato : andare a vedere la getObjectInfo del bean 🡪 cercare se ci sono PROPERTY\_LOOKUP + PROPERTY\_LOOKUP\_FILTERS 🡺 cerco la stringa EG\_TCOD :

Immagine che contiene testo, schermata, software, Pagina Web

Il contenuto generato dall'IA potrebbe non essere corretto.

[02:09:00]

La tabella del paziente è : SA\_PAZIENTE

* Posso filtrare per PZ\_ID o PZ\_COGN : l’id è un VARCHAR2(16)
* Ci sono tantissime colonne : circa 247 + 4 campi di controllo
* Tutti i tab di testata sono memorizzati tutti in questa tabella

Possibilità di modificare l’id di un paziente a mano: parametro aziendale 🡪 da dentro l’azienda la apro in modifica e vo nel tab PARAMETRI 🡪

PAZIENTI 🡪 checkbox “**modifica identificativo paziente**”

Immagine che contiene testo, software, schermata

Il contenuto generato dall'IA potrebbe non essere corretto.

[**02:15:20**]

BEAN COINVOLTI LATO JAVA PER PAZIENTE :sono più di uno : è uno ma c’è una gerarchia:

* PazienteBean : è il cuore di tutto
* Il paziente è poi padre di altri bean : ha un figlio
  + PazienteInteractive 🡪 parte che contiene i controlli solo lato FE (da maschera)

Perché? Siccome il paziente è utilizzato in ambito integrazioni e nel tempo sono state richieste validazioni sulle maschere che non avevano senso per i servizi :

* Si è creato PazienteInteractive per queste validazioni solo sulle maschere
* Il controller PazienteW3 non lavora con PazienteBean ma con PazienteInteractive
* Se devo modificare qualche cosa che non deve impattare i servizi web di integrazione
  + Vado a fare una modifica in PazienteInteractive
  + PazienteInteractive ha le logiche di validazione in più per gestire gli inserimenti da pagina es: “l’ASL di appartenenza non corrisponde alla regione di appartenenza”
  + Altre logiche di pagina es : metodo setCodiceComuneResidenza(String nuovoComune)
    - Sbianca campi precedenti, calcola cittadinanza e regione e altre info a partire dal nuovo comune messo in pagina. Queste logiche sono solo per le attività da pagina di gestione, e non valgono da integrazione: per questo sono state implementate in PazienteInteractive.
* Se invece devo fare una modifica che deve valere sia per i pazienti gestiti da frontend che quelli gestiti da servizi di integrazione 🡪 lavoro su PazienteBean.
* PazienteBean ha come ancestor Paziente.java 🡪 contiene solo getter e setter e definizione delle proprietà private.
  + Colonne base quasi tutte su Paziente
  + Dettagli (liste) : su PazienteBean
  + Paziente 🡪 PazienteBean 🡪 PazienteInteractive

[**02:24:30**]

Col paziente possiamo anche vedere un’altra cosa. Vediamo il suo metodo retrieve() :

la retrieve è diversa da quelle viste finora nel corso , che è come andrebbero fatte.

Quella del paziente : noi facevamo loadKeyFromBean 🡪 il paziente fa mem.setProperty(“ID”, idPaziente). E’ uguale , solo che il paziente è vecchio e non utilizza molte cose che poi sono state “centralizzate” e gestite con una unica chiamata a singolo metodo : es la chiamata a mem.saveToBean() 🡪 nel paziente richiedono un sacco di righe (getPropertyAs….()) una per ciascuna proprietà del bean:

Immagine che contiene testo, schermata, software, numero

Il contenuto generato dall'IA potrebbe non essere corretto.

Recupero dei dettagli (es: medici – retriveMedici()) : uguale a come lo abbiamo visto noi ma c’è un’altra possibilità :

noi facevamo getPersistenceMAnager.find()

Il paziente invece si segna da una parte la getReferenceForeignKey 🡪 perché poi fa una additional condition : recupera solo tra i medici del paziente quelli che sono attivi

Immagine che contiene testo, schermata, Carattere

Il contenuto generato dall'IA potrebbe non essere corretto.

Altro modo più complesso di caricare i dettagli : definendo una classe di query dedicata

Es: retrieveConsensi()

Immagine che contiene testo, software, Pagina Web, Sito Web

Il contenuto generato dall'IA potrebbe non essere corretto.

* Definisco classe query dedicata
* La recupero con QueryFactory.FindQuery
  + CustomQuery queryConsensi = (CustomQuery) QueryFactory.*FindQuery*("sianc.paziente.ConsensiPazienteQuery");
* Le passo filtri con query.setFilter(“nomeFiltro”, valoreFiltro);
* ph.getPersistenceManager().Find(…. Ultimo parametro la classe query);
  + al posto di che solitamente negli esercizi noi valorizzavamo con il campo, nella tabella figlia, che contiene il rif. al padre che fa riferimento al bean padre (FK)

la Find sta in JDBCPersistenceManager 🡪 in base al filter passato come ultimo parametro , il metodo si comporta diversamente se gli passo una query o un ParentReference . di solito si usa:

* ParentReference
* PerparedStatement : statement scritto a mano
* QueryInterface : classe di query.

Immagine che contiene testo, schermata, Carattere

Il contenuto generato dall'IA potrebbe non essere corretto.

[**02:32:00**] inizio pausa [**02:36:00**] – fine pausa [**02:59:00**]

PazienteInteractive utilizzato solo dal controller PazienteW3 (frontend)

if( "$INSERT".equals(getStatus()) 🡪 controllo utile sul controller, per capire se sono in insert o find

I bottoni sono dichiarati nel controller : noi nella JSP wizard avevamo messo il bottone con onclick = doAction(‘CREA\_CORSO’) [codice html]

Invece nel paziente lo definisce il controller lato java come oggetto, button gli dà una label e gli dice l’azione da mandare sulla doAction():

Immagine che contiene testo, Carattere, schermata

Il contenuto generato dall'IA potrebbe non essere corretto.

Poi sulla JSP (anagPrinc.jsp): mette un tag forms:button con attributo buttonId=”BUTTON\_RICALCOLACF”

Immagine che contiene testo, schermata, software, Pagina Web

Il contenuto generato dall'IA potrebbe non essere corretto.

Questo consente di gestire dinamicamente la visibilità sul controller. 🡪 addStatusChecker()

Immagine che contiene testo, schermata, Carattere, numero

Il contenuto generato dall'IA potrebbe non essere corretto.

Il vero valore aggiunto di averlo gestito sul controller è che andando nella profilatura configurabile (sotto menu avanzate), lancio una ricerca per il punto di menu del paziente e poi nel dettaglio : quel bottone lo troverò tra quelli disponibili per decidere se deve essere visibile o no, o meglio creo la chiave di profilatura che poi dovrò impostare :

Immagine che contiene schermata, testo, software, Icona del computer

Il contenuto generato dall'IA potrebbe non essere corretto.

Se voglio dare la possibilità di poter profilare il bottone, devo dichiararlo in questo modo come fatto sul paziente, lato Java sul controller.

Es: il calcolo del codice fiscale 🡪 nella doAction c’è un if -else con tutte le possibili action (dichiarate anche esse come costanti della classe) e per ciascuna viene chiamato un metodo dedicato di appoggio . Per il calcolo CF ciò avviene subordinatamente alla action ACTION\_CALCOLACF 🡪 doCalcolaCodiceFiscale() 🡪

La doCalcolaCodiceFiscale demanda al bean stesso il calcolo del CF. In un controller come faccio ad avere accesso al bean? c’è una property di default messa a disposizione: mainObject

Immagine che contiene testo, schermata, Carattere

Il contenuto generato dall'IA potrebbe non essere corretto.

Guardiamo un po’ alcune validazioni interessanti del paziente.

Stanno tutte in validateCommon di PazienteBean.

Questo genere di validazioni, alcune sono fisse (non c’è modo di evitarle) però alcune invece (es: quella del comune di nascita) le fa solo se decide che deve fare quella validazione : ci sono dei parametri aziendali che consentono di rilassare alcuni controller restrittivi.

Menu : accesso – azienda 🡪 cerco azienda 🡪 tab PARAMETRI 🡪 pazienti 🡪 c’è un link “**controlli restrittivi**” : posso dire quali controlli voglio fare e quali invece voglio rilassare : i controlli restrittivi sono selezionabili tramite una combo. Occhio che alcuni item se li checko disattivo, altri se li checko li attivo : LEGGERE BENE LA DESCRIZIONE

Il PazienteBean legge il parametro dai controlli restrittivi [metodo readControlloRestrittivo], e poi esegue o meno il controllo se quello in particolare è impostato come da fare o da omettere.

Stessa cosa per alcune logiche di autocompletamento come ad esempio il codice comune di residenza 🡪 lui in automatico mi calcola la asl e la regione di appartenenza. Anche questo automatismo è configurabile tramite parametro aziendale :

Immagine che contiene testo, schermata, software, Icona del computer

Il contenuto generato dall'IA potrebbe non essere corretto.

Altro esempio “sovrascrivi i campi calcolati se il campo base viene modificato ” :

Immagine che contiene testo, schermata, software, numero

Il contenuto generato dall'IA potrebbe non essere corretto.

Alcuni invece sono parametri di azienda che decidono se un certo controllo di validazione, quando fallisce, deve dare un errore bloccante o semplicemente un warning. Vedi esempio sotto : quesit due se non sono flaggati, generano errore bloccante. Se seleziono il checkbox del parametro aziendale, allora quello che prima era errore bloccante diventerà un warning.

Immagine che contiene testo, schermata, Carattere

Il contenuto generato dall'IA potrebbe non essere corretto.

Esempio di recupero parametro per sovrascrittura campi calcolati al cambio campo base:

Immagine che contiene testo, schermata, Carattere

Il contenuto generato dall'IA potrebbe non essere corretto.

**Controlli Restrittivi : E’ UN TIPO PARTICOLARE DI PARAMETRO 🡪 PARAMETRO PER DIMENSIONE**

[**03:21:00**]

ANAGRAFICA SOGGETTI : archivio informatico per gestire persone fisiche o giuridiche che a qualsiasi titolo (PAZIENTE ESCLUSO) hanno rapporti con l’ente.

Immagine che contiene testo, schermata, Carattere, numero

Il contenuto generato dall'IA potrebbe non essere corretto.

* Dipendenti
* Medici di medicina generale
  + Medico ricorverato come paziente : esiste sia come soggetto che come pazinete in entrambe le tabelle di DB, sono due cose distinte
* Clienti
* Fornitori
* Etc…

Il ruolo che svolge un soggetto nei confronti dell’ente è determinato dalla configurazione del suo “rapporto” 🡪 associo al soggetto un RAPPORTO. In base al rapporto poi l’applicativo discrimina la tipologia di soggetto.

**I rapporti rientrano nella tipologia di record AZIENDALI** : significa che i record di quella tabella sono specifici per una azienda all’interno dell’installazione di DB di tutta la sardegna ad esempio. Il rapporto è nei confronti di una di quelle aziende censite nel DB multitenant.

Quando associo a un soggetto un rapporto 🡺 lo sto facendo per quella specifica azienda.

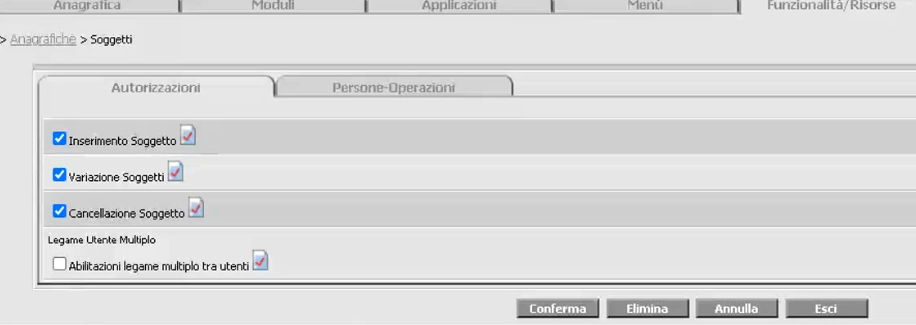
Quando inserisco un nuovo rapporto : IN DB VIENE VALORZZATO IL CAMPO AZINENDA con l’ID dell’azienda🡪 la tabella in db ha una colonna

GESTIONE DB MULTITENANT : quel record sarà visibile solo all’azienda a cui appartiene.

(è come il campo SOC\_SOCIETA\_ID si NETASIAL) 🡪 filtro automatico in maschera per l’azienda di connessione

[**03:28:00**]

**ESERCITAZIONE :** creare medico, creando un rapporto specifico, e poi lo associamo al paziente come scelta e revoca

* Anagrafica : **MENU – SOGGETTI – GESTIONE**
* PAZIENTE E SOGGETTI : la gestione per abilitazione non stanno sono PROFILATURA CONFIGURABILE (sono state fatte prima della introduzione della profilatura configurabile) 🡪 queste chiavi sono state rilasciate fisicamente non sono state fatte con la profilatura dinamica:
* 
* Bisogna andare sotto ANARGAFICHE 🡪 pazienti o soggetti =>> le trovo qua
* Creo il nuovo medico –
  + **La form di gestione (sia paziente che soggetto) NON HA CAMPI OBBLIGATORI : se ne vedo alcuni con sfondo giallo è perché sono stati impostati come tali tramite la profilatura configurabile presso un certo cliente**
  + Unici campi Obbligatori da applicativo :
    - tipo persona fisica o giuridica
    - stato : attivo o disattivo
  + inserita nuova persona : gli assegna un ID e lo vediamo in sola lettura
  + TAB INDIRIZZI : ci andiamo per censire gli indirizzi di un soggetto 🡪 es per una azienda posso mettere l’indirizzo di tipo sede legale, e per ciascun indirizzo una data di attivazione e disattivazione 🡪 per obsolescenza record.
    - Data attivazione utile per l’indirizzo di residenza (c’è un checkbox apposito nella specifica dell’indirizzo) 🡺 per quanto riguarda la residenza, all’interno di un arco temporale può esserci solo un indirizzo di residenza. Quindi se la devo cambiare devo
      * Disattivare il vecchio indirizzo
      * Creare il nuovo indirizzo di residenza flaggando “Residenza”.
    - Date attivazione e disattivazione di due residenze non devono accavallarsi
  + TAB RAPPORTI : è qui che vado a dire che tipo di figura rappresenta il soggetto che ho appena salvato esto definendo. Lo definisco tramite il lookup “tipo rapporto”: mostra tutte le tipologie possibili di rapporti nel mondo AMC e HR + EDF (erogazione diretta farmaco).
    - Tipo rapporto : identifica chi è
    - Data attivazione e disattivazione : il mio soggetto potrà avere N rapporti con la mia azienda di connessione corrente. Per lo stesso rapporto non posso reinserirne uno di stesso tipo a intervallo temporale sovrapposto. Concetto simile a indirizzo di residenza. I rapporti hanno data inizio e fine.

**TABELLE IN DB DELLA PARTI VISTE FINORA** :

* **SA\_PERSONA** : anagrafica del soggetto
  + PF\_ID : è la PK , identificativo del soggetto
  + PF\_TIPO : fisica o giuridica
  + Ci sono i dati di testata anagrafici
  + PF\_RAGSOC : ha senso solo in caso di persona giuridica . se fisica : c’è un automatismo che dentro ci concatena NOME + COGNOME
  + PF\_NORMC, PF\_NORMN,PF\_NORMRS: nome cognome e rag soc hanno questi campi aggiuntivi per togliere doppie e spazi in modo da gestire eventuali errori di battitura. Sul paziente non si usa +
* **SA\_PERS\_IND** : TAB INDIRIZZI DELLA PERSONA – TABELLA DI DETTAGLIO
  + Ha una PK a colonna multipla : PR\_ID + PR\_PROGR
  + PR\_PROGR : progressivo autocalcolato assegnato dal sistema ogni volta che aggiungo un nuovo dettaglio indirizzo a una persona
* **SA\_RAPP\_INDIV** : rapporti individuali – dati messi nel tab “RAPPORTI”
  + RI\_ID : PK gestita come sequence
  + Per filtrare 🡪 RI\_PERSONA : contiene l’id della persona
  + RI\_AZIENDA : è il record che rende il rapporto aziendale ,contiene l’id dell’azienda. **Colonna aziendale**
    - Il record è visibile solo da quell’azienda

**FINE VIDEO 09 : poi finiremo l’esercizio di creazione del medico**