

# Database teoria - 15/9/2023

domenica 22 ottobre 2023 18:55

15/9/2023

## DATABASE

Un DATABASE è un contenitore di dati e può essere relazionale o non relazionale.

Le sono 4 fasi per la creazione di un database: la prima è l'INTERVISTA, ovvero capire le esigenze del cliente.

NB → metafora dell'albero

REACTA → SCHEMA CONCETTUALE → SCHEMA LOGICO → SCHEMA FISICO

Intervista, capire le esigenze del cliente. (con REACTA)

Traduzione schematica (grafica) che restituisce il programma, ovvero la struttura o schema ER (Entità/Relazioni).

Traduzione in SQL.

Schema fisico.

\* Individuare il database bisogna individuare le entità e i loro attributi (caratteristiche). Le entità sono relazionali tra loro. Nella schema ER le entità sono dei rettangoli, mentre le associazioni (relazioni) sono dei rombi. Il sistema del simbolo utilizza un termine che descrive la relazione tra le due entità.

gli attributi sono dei pallini. - mono di fianco alle entità (e delle associazioni, MA SOLO QUELLO MONO A MONO, 1:1).

alt. 1, alt. 2, alt. 3, alt. 4, alt. 5, alt. 6

ENTITÀ 1, ASSOCIAZIONE, ENTITÀ 2

Mezzo è un attributo per il quale è imposta una chiave primaria (CHIAVE PRIMARIA) di identificare un modo univoco ogni istanza dell'entità.

Si rappresenta nell'ER con un pallino e la parola sottolineata.

La chiave candida è un attributo che ha tutte le caratteristiche per diventare chiave primaria.

L'entità è il gruppo generale (STUDENTI) mentre l'istanza dell'entità è l'elemento del gruppo (MARIO, ELENA, LORENZO, ecc...).

Tipi di relazioni tra entità:

- Relazioni 1:1 (uno a uno)
- Relazioni 1:N (uno a molti)
- Relazioni M:N (molti a molti)

Associazioni limitate:

Associazioni multiple (da evitare!!! sempre da convertire):

In semplificare:

Tra due entità forma esiste anche più associazioni:

Anagrafica: insieme di informazioni utili a descrivere una persona o altro (Computer: nome, password, ecc...).

Storica: storia con più di due spaz. temporali differenti.

25/9/2023

## DA ER A LOGICO

Prima traduzione, nella schema logica, le entità sono tabelle:

ogni colonna contiene gli attributi di quella entità. le associazioni possono anche essere tradotte come tabelle, con i suoi eventuali attributi e le chiavi primarie delle entità (entità che ha più multi-comp, meglio usare la 1:N). le chiavi primarie NON possono essere NULL.

Oltre alle chiavi primarie e candidate, esiste anche la CHIAVE ESTERNA.

DOMINIO → insieme di tutti i valori elementari che un attributo può assumere.

CARDINALITÀ → numero di righe; ERMO → attributi - caratteri.

Rappresentazione logica → MODELLO (C, F, Segno, Nome).

Guardare esempi nel quaderno (1:1, 1:N, N:M).

**Selezione:** applicata a una tabella fornisce come risultato l'insieme delle occorrenze che soddisfano la condizione specificata. Il risultato è una nuova tabella che contiene tutte le colonne della tabella di partenza (stesso grado) ma ha cardinalità (numero di righe) minore o al limite uguale.

**Proiezione:** applicata a una tabella fornisce come risultato una nuova tabella che contiene tutte le righe della tabella di partenza (stessa cardinalità), ma con le sole colonne indicate (grado inferiore).

**Congiunzione (join):** si ha quando ci sono due tabelle che sono in relazione. Si crea un'unica tabella che ha → **grado della prima + grado della seconda - elemento in comune**

## Normalizzazione

La normalizzazione è l'insieme di criteri di progettazione di un database relazionale diretto a prevenire l'insorgere di tali anomalie. La normalizzazione è il procedimento che trasforma successivamente le relazioni di partenza suddividendole in altre più piccole aventi lo stesso contenuto di informazione.

Si distinguono tre tipi di anomalie:

- anomalia di inserimento: se nell'inserire un nuovo record in una tabella si è costretti a inserire informazioni già presenti nel DB;

- anomalia di cancellazione: se nel cancellare un record si è costretti a cancellare informazioni che possono essere ancora utili nel DB;
- anomalia di aggiornamento: se dovendo aggiornare un record si è costretti ad aggiornarne molti altri

#### **Prima forma normale**

**Atomicità:** 1 campo deve contenere una e una sola informazione (e il dominio deve essere semplice, ovvero stesso tipo di dato).  
Dominio semplice, atomicità e no multivalore.

#### **Dipendenza funzionale:**

STUDENTE (Matricola, Nome, Telefono, Corso, Voto)

In essa si possono evidenziare le seguenti dipendenze funzionali:

Matricola → Nome

Matricola → Telefono

Matricola, Corso → Voto

Nome ha una dipendenza funzionale da Matricola.

#### **Seconda forma normale**

Una relazione è in seconda forma normale quando è già in prima forma normale (il dominio è atomico).

La seconda forma normale (2FN) si applica alle tabelle che hanno la chiave primaria composta da più attributi (multicampo): è richiesto che tutti gli attributi di una riga dipendano dall'intera chiave primaria e non solo da una parte di essa.

#### **Terza forma normale**

La Terza Forma Normale (3NF) è una forma di normalizzazione dei database che si applica per garantire che:

- La tabella sia già in Seconda Forma Normale (2NF).
- Ogni attributo non chiave sia dipendente direttamente dalla chiave primaria e non da altri attributi non chiave (eliminando le dipendenze transitive).

# Database pratica- 18/10/2023

giovedì 19 ottobre 2023 12:50

Pad Informatica: <https://digipad.app/p/483307/461fb009fcb94>

**XAMPP** è il più popolare ambiente di sviluppo PHP.

L'acronimo **XAMPP** si compone così: **X** sta per Cross-platform, **A** per Apache server, **M** da MySQL, mentre la doppia **P** per PHP e Perl.

Directory **htdocs**: è la radice, la directory principale. All'interno si trovano tutti i progetti. Corrisponde al localhost.

Il file **index.php** (all'interno di **htdocs**) contiene le informazioni sul server.

In fase di avvio:

- **sudo /opt/lampp/manager-linux-x64.run**
- Startare **MySQL Database** e **Apache Web Server**

Sul browser:

- **localhost/dashboard**
- **localhost/phpmyadmin** → gestion grafica del database

Per permettere di creare cartelle all'interno di **htdocs/Galeasso\_ProgInfo5L**, aprire il terminale nella cartella **htdocs** e digitare → **sudo chown federico:federico Galeasso\_ProgInfo5L**

Se apriamo **localhost/Galeasso\_ProgInfo5L** sul browser, vediamo il nostro progetto.

Da la precedenza a **index.html**, quindi apre sempre quel file come principale. Possiamo chiamarlo **home.html** e creare un file **index.php** in modo da aprire il file php come principale.

**Sintassi PHP:**

```
<?php
    echo "Hello world!"
?>
```

## SQL

```
mysql -u root
mysql -u root < ./Progetti_SQL/Galeasso_Orvieto.sql
```

**Non usiamo l'interfaccia grafica perché usa troppe risorse, invece usiamo il terminale perché è più veloce e utilizza meno risorse.**

**Cos'è il DBMS:** è un sistema che permette di gestire i database. Oltre ad essere un software, comprende anche la parte hardware dove gira fisicamente il DBMS.

**Cos'è l'integrità referenziale.** E' un vincolo, un insieme di regole che il DBMS applica a ogni operazione che andiamo a fare. Per preservare i dati correlati ad un'altra tabella. Se non ci sono le relazioni restano dei record orfani. Di default ti blocca, per evitare che vada a fare ricerche di una cosa che non c'è, e che ci siano dati ancora correlati.

Esempio: DB delle fatture --> i dati sono memorizzati in più tabelle a parte. In questo caso conviene alterare l'integrità referenziale di base, perché cancellando la fattura ci serve anche cancellare tutte le sue informazioni (clienti, dati, fornitori).

Si può fare con **ON DELETE CASCADE**. Oppure **ON UPDATE**, in caso di aggiornamento.

Se hai una tabella principale, e due tabelle con le chiavi esterne dentro, se metti **ON DELETE CASCADE** solo su una si blocca perché ha ancora un vincolo referenziale. Se lo metti in entrambe, quando cancelli la tabella principale verranno cancellati di conseguenza tutti gli altri record delle tabelle collegate.

**Indice:** lavora sulla performance del db. Velocizza le ricerche sui campi indicizzati. A volte rallenta perché quando vai a cambiare il dato devi anche cambiare l'indice. Sono tutte tabelle con valori ordinati.

**!!! LA CHIAVE ESTERNA IMPLICA IL VINCOLO DELL'INTEGRITA' REFERENZIALE !!!**

**3 grandi gruppi:** DDL (definire la struttura del database, come il comando create table), DML (Inserire modificare e cancellare i dati), QL (estrarre i dati).

## Integrità dell'entità

Sono vincoli semantici, legati al significato del termine. Non vengono imposti nel DDL

**ADD CONSTRAINT fk\_nomeAz** --> è il vincolo della chiave esterna. E' formato dal nome dell'entità e un numero incrementale.

Semplicemente facendo **FOREIGN KEY (nomeAzienda) REFERENCES Azienda(nomeAzienda)** l'SQL te lo inserisce in automatico, mettendo un nome come ad esempio: **Azienda\_ibfk\_1**, con un numero incrementale.

**DML**--> inserimento, modifica, eliminazione.

Meglio mettere le chiavi e quelle che compongono le chiavi esterne NOT NULL, per non violare l'univocità.

!!!

La **delete** fa una selezione (cancelli i record che vuoi), la **truncate** distrugge la tabella e ricrea la tabella vuota

!!!

## QL

SELECT --> Proiezione (colonne)

WHERE --> Selezione (righe)

COUNT --> Conteggio delle righe

```
SELECT CognomeCli
FROM Clienti
WHERE NomeCli = "Mario"
```

Per selezionare solo ad esempio i cognomi che iniziano con B --> WHERE Cognome LIKE 'B\*';

Per stampare i cognomi che contengano 'rio' --> WHERE Cognome LIKE '%rio%';

Per stampare i cognomi che iniziano o finiscono per 'io' --> WHERE Cognome LIKE '%io' oppure 'io%';

Con underscore \_ al posto della percentuale % si deve avere per forza UN carattere davanti. Per avere ad esempio due caratteri: \_\_ (doppio underscore).

STR\_TO\_DATE(' ',' ') --> formato data

JOIN --> **WHERE Esami.ID\_Esame = Studenti.ID\_Esame**

```
-- 13. Estrarre cognome, nome, e il nome dell'esame degli studenti iscritti all'esame di impianti elettrici
SELECT Cognome, Nome, Nome_Esame
FROM Studenti, Esami
WHERE Esami.ID_Esame = Studenti.ID_Esame
AND (Nome_Esame = 'Sistemi Operativi' OR Nome_Esame = 'Impianti Elettrici');
```

```
-- Estrarre cognome e nome, nome dell'esame e i crediti dei ragazzi iscritti a medicina
SELECT Cognome, Nome, Nome_Esame, Crediti
FROM Studenti, Esami
WHERE Esami.ID_Esame = Studenti.ID_Esame
AND (Corso_Laurea = 'Medicina');
```

```
-- Estrarre cognome, nome, nome esame, e crediti degli esami che hanno i crediti compresi tra 7 e 9
SELECT Cognome, Nome, Nome_Esame, Crediti
FROM Studenti, Esami
WHERE Esami.ID_Esame = Studenti.ID_Esame
AND (Crediti BETWEEN 7 AND 9);
```

## PHP

Programmazione oggetti (OOP).

**!!! this richiama l'istanza (metodi non statici), self richiama la classe (si usa con i metodi statici) !!!**

**!!! I metodi statici non devono essere istanziati, quelli non statici si (costruttore) !!!**

**I metodi statici li usi quando quella funzione non devi usarla o toccarla fuori.**

**Con il PHP possiamo fare le operazioni che facevamo da terminale tramite la nostra pagina web. Creiamo direttamente il DB, lo riempiamo e modifichiamo.**

PHP si attacca al server (localhost), usa quella porta, utente root e password vuota. Gli diremo di usare il db "nome db". Da lì in poi costruiamo le tabelle. Riempiamo i dati. Facciamo le query per eliminare, aggiungere i dati.

Si possono gestire i permessi per gli utenti, ad esempio un utente può aggiungere e non modificare.

**POLIMORFISMO:** capacità di usare la stessa funzione ma che ti dà come return cose diverse, perché si adatta.

## CONVENTION – PROG FINALE

Progetto finale informatica.

Primo sprint: STRUTTURA, già fatto DDL e DML

Secondo sprint: tabella user per l'autenticazione, possibilità di legare con relatori e partecipanti.

La tabella user comprende: amministratore, partecipanti, relatori.

L'utente può essere quindi l'amministratore, il partecipante o il relatore.

L'amministratore crea gli speech.

Il relatore si associa agli speech.

Il partecipante sceglie lo speech a cui partecipare.

Nella login, l'utente si logga come partecipante. E' l'admin che registra il relatore.

Terzo sprint: vetrina, query di selezione per vedere i programmi speech e relatori.

Con sessione autenticata, o senza sessione autenticata.

Con sessione autenticata --> l'utente può iscriversi come partecipante o relatore, e risulterà nella tabella

Quarto sprint: classi personalizzate

## JOIN

cross join, piano cartesiano

inner join, join esplicita, classica

outer join si divide in right e left

left, tabella di sinistra prende tutti i record anche se non ha corrispondenti con la seconda

right, prende tutti i record che hanno corrispondenze

join esplicita, join, join on

se i due campi hanno gli stessi nomi, puoi usare using e tra le tabelle il campo

terzo livello, si usa anche la parola natural. non c'è più l'on e using

join implicita, si usa la select

join fra più tabelle a - b- c:

a con b, risultato di a e b con c

Si fa sempre due a due.

## VISTE

sono tabelle virtuali.

le viste servono per mostrare tutti i dati del db dal momento che ci lavori sopra, sono sempre aggiornate.

Esempio:

```
CREATE VIEW view_palestra AS  
SELECT data_certificato  
FROM certificati  
WHERE data_certificato = '2013-01-11';
```

Definizione:

In SQL, le "viste" (views in inglese) sono query salvate, che possono essere trattate come tabelle virtuali. Quando crei una vista, stai essenzialmente creando una rappresentazione strutturata dei dati presenti in una o più tabelle, ma senza dover memorizzare fisicamente i dati stessi. Le viste possono essere utilizzate per semplificare le query complesse, nascondere la complessità dei dati sottostanti e fornire una visione strutturata e coerente dei dati.



domenica 22 ottobre 2023 18:52

14/9/2023

RIPASSO  
(TCP/IP)

Diagramma delle stack di rete:

- APPLICAZIONE
- TRANSPORT (PDU → SEGMENTO)
- NETWORK (PDU → PACCHETTO)
- FISICO (PDU → FRAME)

Indirizzo IP (32 bit): IPv4: decimale, 4 ottetti da 8 bit (0-255); IPv6: 128 bit, esadecimale, 32 cifre

Indirizzo MAC (48 bit): identifica la scheda di rete

TCP e UDP (datagrammi)

LIVELLO TRASPORTO (pag. 262)

Il livello TRASPORTO lavora con i SOCKET (o numeri di porta).

Diagramma del pacchetto di trasporto:

- DATI (APPLICAZIONE)
- PORTA SOURCE (16 bit) / PORTA DEST (16 bit)
- PAYLOAD

Diagramma del pacchetto di rete:

- IP SOURCE / IP DEST
- PAYLOAD

Diagramma del pacchetto fisico:

- MAC MITT / MAC DEST
- PAYLOAD

Permessione di individuare il giusto percorso a cui inviare il pacchetto di destinazione.

È un'interfaccia tra il livello APPLICAZIONE e il livello TRASPORTO.

non gestite dalla IANA

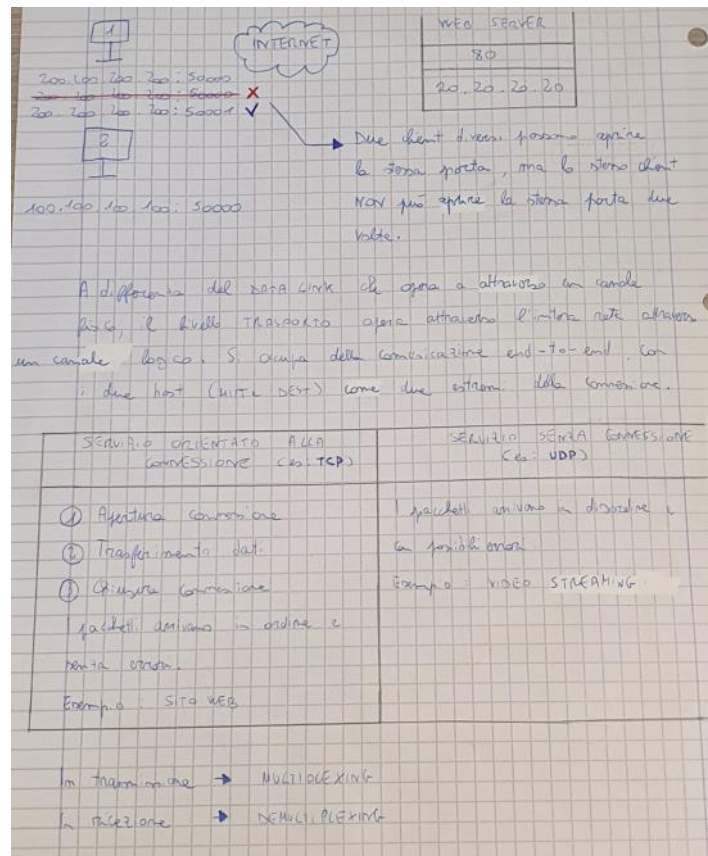
Numero di porte possibili →  $2^{16} - 1 = 65535$

1°: 0 - 1023 → PORTE NOTE (WELL KNOWN PORTS), assegnate al S.O.

2°: 1024 - 49151 → PORTE REGISTRATE (accede a pagamento con la IANA)

3°: 49152 - 65535 → PORTE DINAMICHE o PRIVATE (non quelle utilizzate dal S.O. per porte registrate)

Porta 80: http



- \* CONNECTIONLESS (Non è con affidabile)
- \* CONNECTION ORIENTED (con affidabile)

Protocollo DNS → fa la traduzione tra il nome di dominio e l'indirizzo IP.  
 ↳ 8 byte

- \* Datagramma UDP: Numero porta sorgente/destinatario, length, checksum, data. (da pag 273, figura pag 274. 3.2)

⚠️ PROBLEMI: Request Indication, Response, Confirm. Sono delle funzioni che permettono di capire se i pacchetti:

- \* UDP-Lite: Numero porta sorgente/destinatario, checksum, lunghezza, data. A differenza del datagramma UDP, UDP-Lite monitora gli errori. Per esempio, in una comunicazione VoIP che riceve molte frammenti a capire il contenuto.

(da pag 277) ↳ 20 byte

- \* Sequenza TCP: Numero porta sorgente/destinatario, sequenza number, ... → (vedi pag 275)

# FTP - 16/10/2023

lunedì 16 ottobre 2023 14:19

**FTP** (*File Transfert Protocol*) serve a trasferire file.

Su Packet Tracer:

- **ftp + indirizzo del server**: serve a collegare un pc al server. In seguito bisogna fare il login dell'utente.
- **get + nome file**: inserisce il file nel server.
- **put + nome file**: prende il file dal server.
- **quit**: serve per uscire dal server.
- **?**: mostra l'elenco dei comandi.

La connessione client-server può avvenire secondo due modalità:

- **Active Mode**
- **Passive Mode**

*Esercizi svolti → cartella Sistemi in Galeasso\_5L*



# DNS 19/10/2023

giovedì 19 ottobre 2023 08:20

Sul libro del 4° anno → da pagina 329

Esercizi svolti → cartella Sistemi in Galeasso\_5L

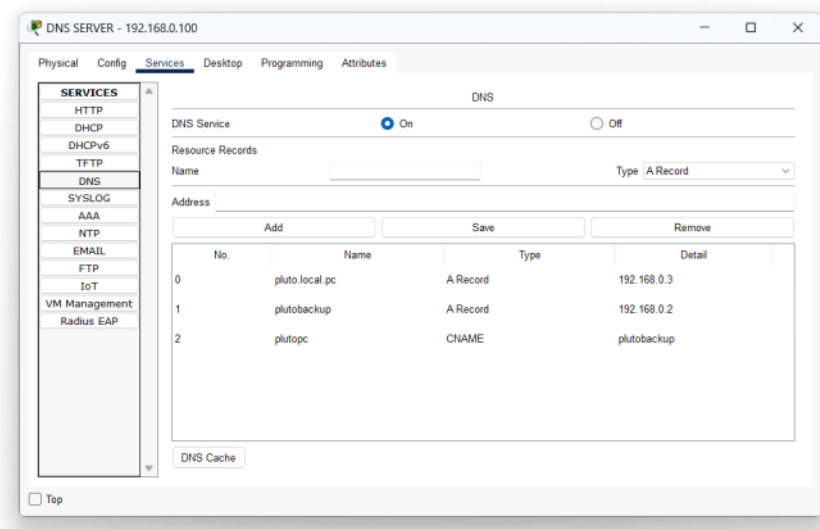
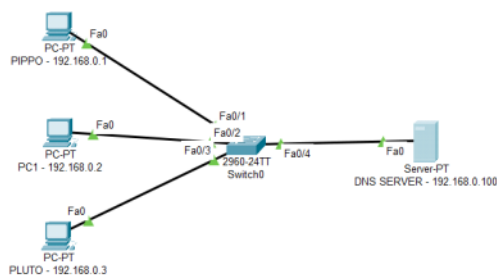
Il **DNS** (Domain Name System) un protocollo che traduce il nome del dominio in un indirizzo IP (o viceversa).

In una scheda di rete solitamente vengono impostati due DNS server (o Name Server). Uno è quello di default e l'altro è quello secondario.

L'ISP fornisce il DNS Server di default.

Su **Packet Tracer**:

- Nel server, andare nella sezione **Services**, poi nella sezione **DNS**.
- Impostare **DNS Services** su **On**
- **Resources Records** → tengono in memoria i nomi di dominio.
  - **Name** → nome del dominio da inserire
  - **Type**:
    - A Record (IPv4)
    - Se selezioniamo CNAME, possiamo dare un alias al PC.
- Sui PC → Inserire nel DNS Server (nella sezione IP Config) l'indirizzo IP del server



Il DNS è formato da **3** componenti principali:

- **Domain Name System**
  - **Domini radice** → domini di primi livello (**TLD**)
  - **Domini intermedi** → domini che hanno altri sottodomini
  - **Domini foglia** → contengono solo host e sono privi di altri sottodomini
- **Name Server**
- **Resolver**

**Root nameserver** → vanno dalla A alla M e sono sempre attivi sulla rete.

**nslookup** [www.denina.it](http://www.denina.it) → ti fornisce l'indirizzo IP associato al sito web. L'accesso tramite indirizzo IP può essere bloccato per motivi di sicurezza.

```
Prompt dei comandi
Microsoft Windows [Versione 10.0.22621.2428]
(c) Microsoft Corporation. Tutti i diritti riservati.

C:\Users\feder>cd
C:\Users\feder>

C:\Users\feder>nslookup www.denina.it
Server: UnKnown
Address: 192.168.10.250

Risposta da un server non autorevole:
Nome:   pvw-ssl.spaggiari.eu
Address: 217.27.72.228
Aliases: www.denina.it

C:\Users\feder>
```

Per **modificare** un file index.html andare su Services, HTTP e cliccare, sulla riga del file index.html, il pulsante **(edit)**.

# HTTP e WWW - 26/10/2023

giovedì 26 ottobre 2023 08:49

Sul libro del 4° anno → da pagina 371

Il protocollo **HTTP** (*HyperText Transfer Protocol*) serve per la connessione ad un Web Server (pagina web). Il client, per comunicare, usa il browser.

E' un protocollo **stateless** (senza memoria).

**Non** è un protocollo **sicuro** perché i dati passano in chiaro.

Si trova sulla porta **80**.

A cosa servono i **cookie**? Servono a tenere traccia di ciò che succede e salvano le informazioni (es: la ricerca che abbiamo fatto, login, preferenze sull'aspetto, lingua, ecc...).

**TABELLA 1** HTTP identifica le risorse del WWW mediante un indirizzo simbolico: URL

http://	www.	azienda.com	/news/
indica al browser quale protocollo deve essere usato	identifica il nome di una specifica macchina (il web server)	rappresenta l'entità di dominio (domain entity) del sito web	identifica la cartella dove si trova la pagina web sul server. Se non viene specificato nulla, il browser carica la pagina web di default presente sul server

A cosa serve il **proxy**? Si occupa di fare la richiesta al server e si occupa di offrire la risposta al client. Permette di memorizzare nella cache alcune informazioni in modo da velocizzare le ricerche. Permette anche di bloccare alcune richieste. Prima di fornire la pagina, il proxy fa una richiesta all'**header** e se la pagina è modificata fornisce la pagina aggiornata, altrimenti mostra quella salvata in cache.

# Posta elettronica – 9/11/2023

giovedì 9 novembre 2023 09:41

# VLAN – 18/12/2023

lunedì 18 dicembre 2023 13:05

Sullo switch (0 e 4096 sono riservati, da 1 a 4095 si possono usare):

- CLI --> enable, conf t, vlan Numero\_VLAN (1 è la VLAN base), name Nome\_VLAN, exit
- Parte grafica --> VLAN Database

Per vedere le vlan --> show vlan brief

Per associare la porta alla VLAN:

- CLI --> conf t, interface fastEthernet Numero\_Porta, switchport access vlan Numero\_VLAN
- Parte grafica --> in config sulla porta che vogliamo (es: FastEthernet0/1), modificare la VLAN

trunk --> abilita il passaggio di pacchetti tra VLAN differenti, access no

Router On Stick:

```
interface gigabitEthernet 0/2.10
encapsulation dot1q 10
ip address 192.168.10.254 255.255.255.0
Exit
```

Gli switch di terzo livello gestiscono anche gli indirizzi IP, e fungono quindi come router.

Su PKT --> Switch 3560 (terzo livello)

```
enable
conf t
interface vlan 10
ip address 192.168.10.254 255.255.255.0
exit
-----
ip routing
```

```
enable
conf t
interface gigabitEthernet 0/2
no switchport (fa in modo che la porta non sia più la porta di uno switch)
Exit
```

Se si fa un ping in broadcast, il router cambia l'interfaccia della vlan e la invia.

# STP – 1/2/2023

giovedì 1 febbraio 2024 08:19

**STP** serve ad evitare che rimangano dei pacchetti che girano all'infinito (loop). Vengono chiamate le **broadcast storm**.

Si possono collegare 3 switch tra di loro in modo che se si "rompe" un collegamento abbiamo ancora on gli altri due switch.

Prevede che una porta venga disabilitata: la porta è ancora attiva ma i pacchetti non passano. Viene stabilita una gerarchia e vengono stabiliti i collegamenti.

Se si "rompe" un collegamento, la porta verrà riattivata per far passare i pacchetti, dato sul collegamento rotto non possono passare.

NB DOMINIO DI COLLISIONE E DI BROADCAST

BRIDGE vs SWITCH:

Il bridge ha 3 porte:

- WAN: collegamento con l'esterno
- 
- A e B sono due segmenti. La rete viene divisa in due segmenti, e i dispositivi sono connessi tramite bus condiviso.

Il bridge mantiene la mac address table, con tutti gli indirizzi mac del segmento A e B.

Se il destinatario si trova in A, lo scarta perchè sicuramente è già stato ricevuto. Se si trova su B, confronta l'indirizzo e invia il messaggio. Se non si trova ne in A e ne in B lo invia sulla porta WAN.

BPDU: individuano i bridge/switch impostati come root. Quello con bridge id più basso sarà lo switch root. Inoltre calcolano i percorsi. Il messaggio di hello/eco non viene mandato dal momento che lo switch ha bridge id alto (esempio. Sw con bride id 3 e altro sw con bridge id 4, dopo essersi scambiati i messaggi, 4 smette di inviare il messaggio di eco).

BRIDGE ID = CODICE (uguale per tutti gli sw) + MAC VLAN DI DEFAULT (1).

Per vedere BRIDGE ID: lente di ingrandimento sullo switch e poi port status summary table.

CLI --> show version

Show spanning-tree --> fa vedere il bridgeid

Il bridge id serve anche per eleggere lo switch designato (per ogni lan è lo switch più vicino alla root, dove passano tutte le comunicazione della lan).

Inoltre, permette di scegliere la root port, ovvero il percorso migliore.

STATI DELLE PORTE:

- disabled
- blocking (blocco)
- listening (ascolto)
- learning (apprendimento)
- forwarding (trasmissione)

Per disabilitare una porta: conf t, interface fastethernet 0/2, shutdown.

# Firma digitale – 1/2/2023

giovedì 1 febbraio 2024 09:14

Da pag 22 e Digipad

La firma digitale è un codice, un'impronta. E' una sequenza di bit che permette di indentificare la paternità di chi ha inviato il documento.

Garantisce: Validità, veridicità e paternità.

Integrità, autenticità e non ripudiabilità.

Una volta c'era la firma autografa (in comune si metteva un timbro sulla firma).

Chiavi asimmetriche

Due chiavi: pubblica e privata.

Certificato di autenticazione CNS: è utilizzato per verificare l'identità nei processi pubblici (esempio spid). Si trova su smart card o server.

Certificato di sottoscrizione: certificato di firma, depositato su un server online o smartcard

Software di firma per firmare: GoSign.

Formati utilizzati per produrre file firmati digitalmente:

- Pkcs#7 (p7m): amministrazione pubblica
- PDF
- XML

Altri enti certificatori:

- PEC
- SPID
- CNS

# Firewall, Proxy, ACL - 19/2/2024

lunedì 19 febbraio 2024 12:28

Appunti su Digipad e sul libro a pagina 44.

Wildcard --> host 1, rete 0 (inverso della maschera di rete, perché non si tratta di un singolo host ma di tutta la rete)

Maschera di rete --> host 0, rete 1

1. Le ACL standard vengono utilizzate per bloccare o permettere il traffico da una rete o da un host specifico o per negare una suite di protocolli. L'aspetto fondamentale delle ACL standard è che il controllo viene esclusivamente effettuato sull'indirizzo sorgente.  
1-99
2. Le ACL estese forniscono una maggiore flessibilità e controllo poiché possono effettuare il controllo non solo sull'indirizzo del mittente, ma anche su quello del destinatario, su uno specifico protocollo, sul numero di porta o su altri parametri  
100-199

IN:

- 1) Viene valutata l'ACL
- 2) Dopo viene analizzata la tabella di routing (effettua il routing)

OUT:

- 1) Effettua il routing
- 2) Viene valutata l'ACL

OGNI PORTA DEL ROUTER PUÒ AVERE 2 ACL (UNA IN INGRESSO IN E UNA IN USCITA OUT)

L'ACL Standard va messa, di base, vicino alla destinazione.

L'ACL Estesa va messa, di base, vicino alla sorgente (così si evita che il pacchetto giri e venga scartato alla fine).

Vanno messe prima le ACL generali e poi quelle più specifiche.

enable ----> show access-lists --> elenco delle acl che ci sono  
show ip interface --> elenco di tutte le interfacce con le acl settate

-----

Router#

Router#

Router#conf t

Enter configuration commands, one per line. End with CNTL/Z.

Router(config)#no access

Router(config)#interface gig

Router(config)#interface gigabitEthernet 0/1

Router(config-if)#no ip access-group 1 out

Router(config-if)#exit

Router(config)#interface gigabitEthernet 0/2

Router(config-if)#no ip access-group 1 out

Router(config-if)#exit

Router(config)#no ip access-l

Router(config)#no ip access-list 1

^

% Invalid input detected at '^' marker.

Router(config)#no access-list 1

Router(config)#exit

Router#

%SYS-5-CONFIG\_I: Configured from console by console



```

Router#
Router#
Router#show access-lists
Router#conf t
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#
Router(config)#
Router(config)#
Router(config)#access-list 1 deny 192.168.3.0 0.0.0.255
Router(config)#access-list 1 permit any
Router(config)#access-list 2 deny 192.168.1.0 0.0.0.255
Router(config)#access-list 2 permit any
Router(config)#exit
Router#
%SYS-5-CONFIG_I: Configured from console by console
sh
% Incomplete command.
Router#
Router#
Router#show acces
Router#show access-lists
Standard IP access list 1
10 deny 192.168.3.0 0.0.0.255
20 permit any
Standard IP access list 2
10 deny 192.168.1.0 0.0.0.255
20 permit any

Router#
Router#
Router#
Router#
Router#
Router#conf t
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#interface gi
Router(config)#interface gigabitEthernet 0/2
Router(config-if)#ip access-group 1 out
Router(config-if)#exit
Router(config)#interface gigabitEthernet 0/1
Router(config-if)#ip access-group 2 out
Router(config-if)#
Router(config-if)#
Router(config-if)#
Router(config-if)#exit
Router(config)#show ip interface gi
Router(config)#show ip interface giga
Router(config)#exit
Router#
Router#
Router#show ip interface giga
Router#show ip interface gigabitEthernet 0/2
....
Router#

```



Sul libro da pag 50.

Cosa fa il router NAT? Mantiene in memoria una tabella,

LAN --> privato  
WAN --> pubblico

Richiesta, DNS fa la traduzione da testo a IP, invio del pacchetto al router, il NAT si memorizza una tabella con IP di chi ha fatto la richiesta e IP di dove deve andare il pacchetto. Per fare uscire una richiesta deve aprire una porta, perché deve gestire più richieste. Successivamente sta in ascolto in attesa della risposta dal server di destinazione. Successivamente verrà inviato all'indirizzo del NAT il pacchetto e dato che il NAT conosce l'indirizzo della sorgente, cambierà l'indirizzo per inviarlo alla sorgente.

PROBLEMA: Può aprire una sola porta verso la destinazione per la stessa rete, rapporto 1:1.

SOLUZIONE: Evoluzione del NAT, piuttosto che gestire gli indirizzi IP, gestisce le porte. Più porte verso il server di destinazione. Si chiama il PAT, rapporto 1:N. Mette a disposizione un campo di 16 bit (65536 porte possibili) per aprire più porte verso la stessa destinazione.

3 funzionalità:

1. Static NAT
2. Dynamic NAT (funzionalità dinamica, permette di smistare su indirizzi pubblici differenti)
3. Port Address Translation (PAT)

NB --> IPv6:

128 bit

1 cifra es --> 4 bit

32 cifre esadecimali

Non c'è distinzione tra IP pubblici e privati. Le cifre non hanno tutte lo stesso significato, ad esempio le ultime fanno riferimento al dispositivo.

Al giorno d'oggi c'è il problema della comunicazione tra IPv4 e IPv6. Per risolvere:

1. Dual-stack: non risolve il problema, perché mancano indirizzi IPv4.
2. Conversion:
3. Tunneling per IPv6: il pacchetto viene incapsulato.

Comandi:

```
router #show ip nat translations
router #show ip nat statistics
router #debug ip nat detailed
router #clear ip nat translation
```

## Sequenza operazioni configurazione NAT statico

1. Configurare le interfacce (assegnare IP-privato, subnetmask, default gateway)
2. Indicare se l'interfaccia inside NAT o outside NAT (ip nat in – ip nat out)
3. Assegnare l'indirizzo statico pubblico (ip nat inside source static 192.168.1.1 209.165.200.254), questo comando indica al router che l'indirizzo privato 192.168.1.1 deve essere sostituito con quello pubblico 209.165.200.254 quando il relativo host invia pacchetti verso l'esterno.

# DMZ - 22/2/2024

giovedì 22 febbraio 2024 09:38

Da pag 56.

Zona demilitarizzata, dove il traffico WAN e LAN sono controllati e limitati.

Ne esistono di due tipi:

1. Vicolo cieco: 1 solo firewall. Dalla LAN alla DMZ può tornare indietro. Un utente esterno può andare fino alla DMZ e tornare solo indietro verso l'esterno, non sulla LAN. Vengono messi solo servizi di front-end.
2. Zona cuscinetto: nel caso di più firewall. Protezione maggiore, svantaggio più costo.

# VPN - 29/2/2024

giovedì 29 febbraio 2024 08:26

Da pag 84 e su Digipad.

Collegare due reti:

- Collegare un cavo fisico: modo più sicuro.  
Problemi:
  - Il costo. (Es.: collegare la bidelleria di verzuolo al denina)
  - Lavori in corso.
- Internet  
Vantaggi:
  - Rete già esistente
  - Apparecchiature già esistenti

La VPN permette di avere un traffico privato (Es.: tutte le LAN delle sedi Unicredit collegate tramite VPN alla WAN. Fisicamente in sedi separate, ma logicamente tutte sotto la stessa rete privata). Il pacchetto cifrato passerà all'interno della rete pubblica, quando arriverà a destinazione verrà decifrato dal destinatario.

Nella VPN ritroviamo nuovamente il tunneling (NB: IPv4 e IPv6).

Due tipi di VPN:

- **Remote-access VPN**  
Lato client: serve il software VPN.  
Lato server: serve il router VPN (NAS, Network Access Server).  
Tra di essi c'è il collegamento VPN, ed è come se facesse parte della rete.  
Questo è utile ad esempio per lo smartworking, sito della banca.
- **Site-to-site VPN**  
Serve per collegare due intere reti (Es.: intera rete di verzuolo con quella del Denina).  
Ci sono due tipi:
  - Intranet-based:  
Solo LAN che appartengono alla mia azienda
  - Extranet-based:  
Se ci sono LAN di appartenenze diverse, quindi un'azienda che ha un rapporto con un'altra azienda.

Il NAS permette:

- Authentication: conferma chi sei.
- Authorization: vengono gestite le autorizzazioni per i vari utenti. (Un utente ha permessi differenti rispetto ad altri utenti.)
- Accounting: tiene traccia di ciò che fai quando sei loggato.

## SICUREZZA NELLE VPN

- Autenticazione: permette l'autenticazione tramite token, qr-code, otp.
- Cifratura: può avvenire attraverso diversi algoritmi di cifratura (3-DES).
- Tunneling: il pacchetto viene incapsulato. Le VPN possono essere il modalità trasporto o tunneling.

Tra livello applicazione e trasporto c'è il livello sicurezza (due certificati: SSL/TLS).

---

**NB PACKET TRACER: Il primo pacchetto va perso, per la richiesta arp, così il router conosce l'indirizzo mac address e può inviarlo a destinatario**

---

**IPsec VPN** a pag 92 e su Digipad

Lavoro in modalità trasporto e tunneling.

E' un insieme di protocolli, funziona sia con Site to site che con Remote access.

Ci sono 3 protocolli principali:

- AH (Authentication Header): garantisce autenticazione e integrità ma non confidenzialità.
- ESP (Encapsulating Security Payload): fornisce autenticazione integrità e confidenzialità.
- IKE (Internet Key Exchange): serve per lo scambio delle chiavi.

Connessione logica --> è detta Security Association, servono per inviare i metadati, ovvero i parametri di sicurezza utili a creare la connessione VPN (algoritmi di sicurezza, ecc...). Sono unidirezionali, quindi ne serve una per l'invio e uno per la risposta. Successivamente si può creare la connessione con IPsec.

SAD --> db con tutte le SA aperte

SPD --> db con tutte le ACL

NB!!!

AH autentica l'intero pacchetto IP, ad eccezione dei campi variabili dell'header.

ESP autentica solo il payload, e non l'header.

Il campo più importante è ISP (Security Parameters Index): identifica in modo univoco la Security Association utilizzata.

IKE utilizza due fasi:

- 1) Viene instaurata la SA (IKE SA)
- 2) Vengono stabilite le due chiavi da utilizzare.

---

**SSL/TLS** da pag 96 e su Digipad

Da rivedere HANDSHAKE (three way handshake, tcp)

Lavorano a livello trasporto (sessione), non sono compatibili, utilizzano solo Remote access (quindi Client --> Server).

E' composto da due livelli:

- Handshake Protocol: serve a concordarsi con il destinatario sui parametri di crittografia, permette anche l'autenticazione
- Record Protocol: serve a cifrare e incapsulare i pacchetti.

Il server invia al client il certificato firmato dalla Certification Authority: il client confronterà con la firma digitale per verificare se è valida.

4 passi:

- 1) Client --> Server: vengono inviati gli algoritmi di crittografia, e viene inviata anche la pre master key, che serve a creare la chiave privata
- 2) Server --> Client: il server invia il certificato digitale, sceglie gli algoritmi, la pre master key, e la richiesta del certificato.
- 1) Client --> Server: il client verifica il certificato del server. Se è valido, invia al server il certificato e la pre master key cifrata con la chiave pubblica del server.
- 2) Server --> Client: verifica del certificato.

---

Le VPN possono essere:

- Trusted VPN: è l'ISP che garantisce il percorso dei pacchetti. I pacchetti viaggiano in chiaro. I dati possono essere catturati in caso di attacco. Non è sicura.
  - Secure VPN: i pacchetti vengono cifrati, è sicuro.
  - Hybrid VPN: per una parte del percorso viene affidato all'ISP, per un'altra parte del percorso i pacchetti vengono cifrati. E' un misto tra i primi due.
-

# Sicurezza reti wireless- 21/3/2024

giovedì 21 marzo 2024 08:25

Da pag 130 su libro e appunti su Digiapad.

Per proteggere una rete wireless bisogna utilizzare una VPN

Possibili attacchi:

- Sniffing
- Access Point Rouge (APR): come se io collego un access point alla rete, per aprire una porta verso il mondo esterno
- Spoofing: Sostituzione del SID (Falsificazione identità modificando il SID)
- Brute Force

Wardriver: vanno in giro a cercare se ci sono reti libere.

## Crittografia

Si cifra dal wireless terminal (telefono, pc) all'access point. Non tra mittente e destinatario. Solo nel tratto wireless, poi dall'access point in poi il pacchetto passerà in chiaro sul collegamento fisico ethernet.

### Tipi di crittografia:

- **Crittografia WEP (Wired Equivalent Privacy)**, primi anni 2000  
A chiave simmetrica a flusso, crittografia base. Algoritmo RC4, viene cifrato solo il payload del frame.  
**NB**  
A **flusso**: si tratta ogni singolo bit; a **matrice (blocchi)**: vengono tenuti in considerazione blocchi di bit (es: 64 bit).  
*Problemi:*
  - Vettore di inizializzazione in chiaro e piccole dimensioni.
  - La chiave rimane sempre la stessa (Brute Force!!!).
- **Crittografia TKIP (Temporal Key Integrity Protocol)**  
Evoluzione del WEP.  
Chiave temporale a 128 bit.  
Vettore di inizializzazione a 128 bit.
- **Crittografia AES (Advanced Encryption Standard)**  
Cifrario a blocchi, molto complesso.  
Non tutti gli access point lo supportano.
- **Crittografia WPA (Wi-Fi Protected Access)**  
WPA 1.0, WPA 2.0.  
WPA 2.0 - Personal (utilizza una PSK, chiave), WPA 2.0 - Enterprise (Utilizza un server di autenticazione, RADIUS).  
WPA 2.0 - Enterprise ha aggiunto la possibilità di vedere le richieste, o vedere le persone che in quella determinata ora hanno visualizzato il sito.  
WPA3 --> 2018. Inventato per risolvere l'attacco KRACK in WPA 2.0. E' stata aggiunta anche la possibilità di fare la connessione di un dispositivo tramite QR Code (easy connect).  
WPA3-Personal: garantisce la sicurezza tramite protocollo SAE (scambio di chiavi peer-to-peer, senza la certification authority, mittente e destinatario gestiscono le chiavi).  
WPA3-Enterprise crittografia a 192bit.

### Autenticazione WPA Aziendali

802.1x e protocollo EAP. Permette di autenticare i dispositivi per la parte cablata e wireless. Permette solo il passaggio di pacchetti autenticati.

Serve un server RADIUS che sfrutta EAP e 802.1x.

Utilizzare il MAC Address non è sicuro perché si può sostituire/mascherare.

!!!Leggere normativa wireless su Digiapad!!!



# Strutture di rete (CLOUD) - 8/4/2024

lunedì 8 aprile 2024 13:22

Sul libro da pagina 188.

# Virtualizzazione - 11/4/2024

giovedì 11 aprile 2024 09:24

Sul libro da pagina 198 e su Digipad.

# Cloud Computing - 15/4/2024

lunedì 15 aprile 2024 13:23

Sul libro da pagina 207 e su Digipad.

# Reti IP Mobili - 29/4/2024

lunedì 29 aprile 2024 13:32

Sul libro da pagina 149 e su Digipad.

# Reti cellulari - 6/5/2024

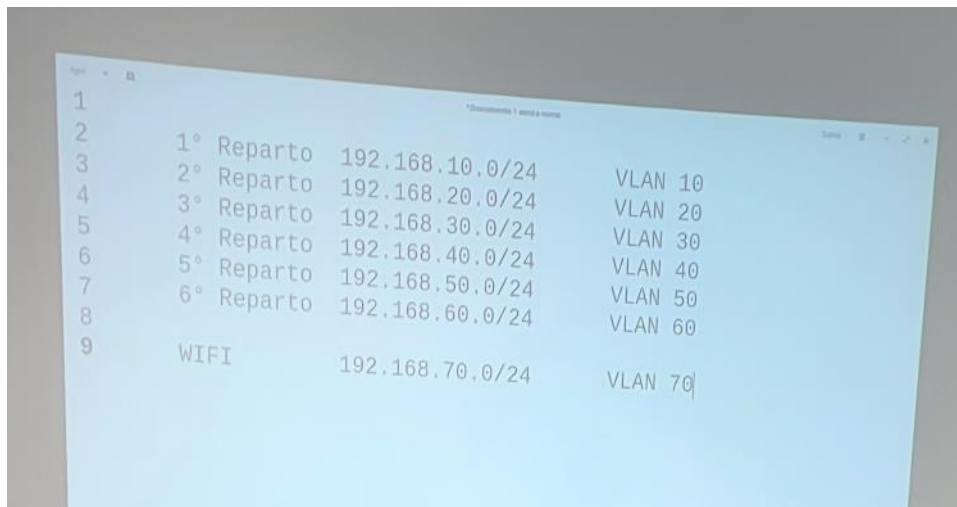
lunedì 6 maggio 2024

13:38

# Prova d'esame

lunedì 20 maggio 2024 14:10

- Salvare i dati su un DB locale va bene, ma sarebbe meglio salvarli attraverso un canale VPN su un DB remoto; in questo modo, in caso di problemi (es: incendio sul DB locale), i dati resteranno integri.
- Non fare subnetting, cidr e vlsm su indirizzi privati (es: rete locale aziendale), solo se richiesto.



1		
2	1° Reparto	192.168.10.0/24
3	2° Reparto	192.168.20.0/24
4	3° Reparto	192.168.30.0/24
5	4° Reparto	192.168.40.0/24
6	5° Reparto	192.168.50.0/24
7	6° Reparto	192.168.60.0/24
8		
9	WIFI	192.168.70.0/24

- MAC Filtering (problema: MAC Spoofing), oppure IEEE con protocollo EAP (Autenticazione reciproca).

# Android Studio 18/9/2023

domenica 22 ottobre 2023 18:07

Su **Android**: si compila su **Android Studio (Java)** e crea il file **APK**.

Su **Apple**: si compila su **XCode (Swift)** e crea il file **IPA**.

**Java** gira su **android nativo**, file **apk**.

Su Android Studio ci sono decine di cartelle contenenti centinaia di file.

I **toast** servono in fase di debug.

Una volta creato il file apk, bisogna collegare un dispositivo android al pc tramite cavo usb. Quando facciamo il "play" vedremo la schermata della nostra applicazione sul dispositivo collegato.

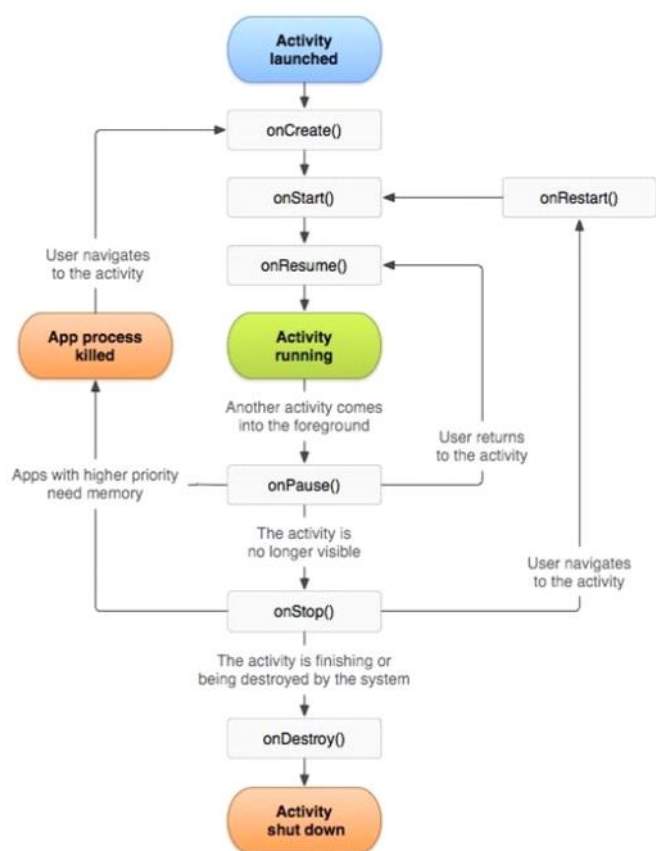
Se non si ha a disposizione un dispositivo android è possibile visualizzare la nostra applicazione sull'emulatore che ci fornisce android studio. Il problema è che occupa le risorse del pc (ram, disco, ecc...).

La sezione **SDK** mostra gli SDK (*versioni android*) installati.

La finestra si chiama **activity**.

In fase di creazione dell'app si deve scegliere l'**API Level**.

**Ciclo di vita dell'activity:**



In bianco sono i **metodi**. Prima di che l'applicazione si apra, vengono richiamati i metodi **onCreate()**, **onStart()** e **onResume()**.

Cartelle su Android Studio:

- Cartella **manifests**
  - Contiene il file **AndroidManifest.xml**, che spiega com'è fatta l'applicazione. All'interno troviamo le **permission**: sono dei permessi ad esempio per mandare le mail, effettuare le chiamate, fotocamera, microfono, file system, ecc.
- Cartella **java**
  - Contiene il progetto, il **package** (es: *com.example.galeasso\_airplanemode*).
- Cartella **res**: contiene le risorse del progetto
  - Cartella **drawable**: contiene i componenti grafici.
  - Cartella **layout**: contiene i layout delle activity. Ogni activity ha il suo layout.
  - Cartella **values**: contiene tutte le stringhe. Il file **strings.xml** (in inglese di default) contiene le regole per evitare di avere ridondanza del codice (es: chiamate a funzione). Ci saranno anche i file **stringsIT.xml**, **stringsEN.xml**, **stringsFR.xml**, che contengono i valori in base alla lingua.

**intent** → **<intent-filter>** → servono per passare da un'activity all'altra. Possono essere impliciti o espliciti.

L'activity principale contiene questi due tag all'interno del tag **<intent-filter>**



- `<action android:name="android.intent.action.MAIN">`
- `<category android:name="android.intent.category.LAUNCHER">`

Qualunque componente grafico deriva dalla classe **View** (es: TextView).

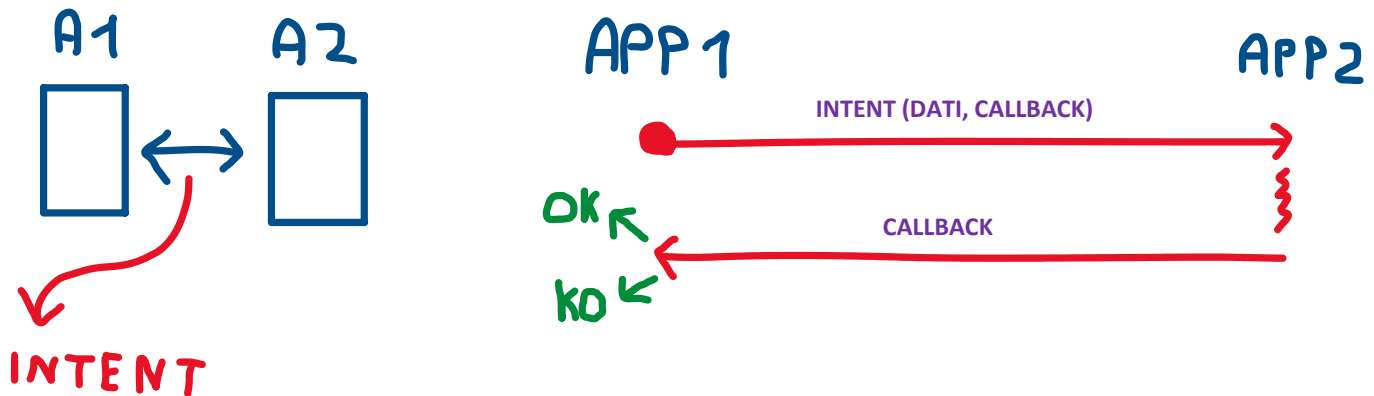
`@+id/nome` → id univoco di ogni classe

`findViewById()` → equivalente del `getElementById`

`@Override` → sovrascrive

In Android Studio → **programmazione multi-thread**

`Intent.ACTION_DIAL` → `intent` è la classe, `ACTION_DIAL` è la costante (valore booleano)



`start activity` → fa partire la seconda activity e non si preoccupa di ciò che succede

`start activity for result` → fa partire la seconda activity e gli dà la callback da chiamare quando finisce

`Log.d("Tag", "Messaggio" + variabile)` → `Log` è una classe, `d` (debug) è un metodo statico. Richiede due stringhe.

`@Override`

`Protected void onCreate(Bundle savedInstanceState){}` →

- **protected**: Questa parola chiave indica che il metodo `onCreate` è protetto e può essere accessibile solo all'interno della classe stessa o dalle sottoclassi.
- **void**: Questa parola chiave indica che il metodo `onCreate` non restituirà un valore (cioè, non restituirà nulla).
- **onCreate**: Questo è il nome del metodo. Il metodo `onCreate` è uno dei metodi fondamentali di una classe Activity in Android. Viene chiamato quando l'activity viene creata.
- **(Bundle savedInstanceState)**: Questo è il parametro del metodo. Accetta un oggetto Bundle chiamato `savedInstanceState`. Un oggetto Bundle è utilizzato per passare dati tra le diverse attività dell'applicazione. Nel contesto di `onCreate`, `savedInstanceState` può contenere dati che sono stati salvati quando l'activity è stata precedentemente distrutta (ad esempio, a causa di una rotazione dello schermo o di un'altra interruzione). Questi dati possono essere utilizzati per ripristinare lo stato precedente dell'activity.

**Interfaccia** → permette di interagire con l'esterno

**Implements** → l'interfaccia non è istanziabile. Da corpo ad un metodo esterno astratto.

Per usare il `clickListener`: **new, this / xml, campo**.

Pag 400

**Permission**: sono i permessi per far funzionare l'applicazione (mandare le mail, effettuare le chiamate, fotocamera, microfono, file system, ecc...)

Esempio: `ACCESS_WIFI_STATE`

Quando scarichi un'applicazione, bisogna accettare le permission.

Su iOS scarichi applicazioni solo dall'AppStore, su Android puoi scaricare da PlayStore o anche da terze parti (apk).

Le permission possono anche essere richieste a runtime (consenti sempre, solo una volta, non consentire).

Le activity comunicano tra di loro con l'intent. Esempio: pagina Login e pagina Benvenuto. Login → Username: utente | Password: 1234 ; Benvenuto: Benvenuto, utente! "utente" viene passato nell'intent attraverso gli extra.

Se fai la richiesta senza permission → eccezione (non funziona)

Esempio: Bluetooth per funzionare deve avere la posizione. Se faccio la richiesta della permission del Bluetooth senza avere fatto quella per la posizione, non funziona

(eccezione).

L'applicazione deve essere firmata.

Se (in fase di test) installiamo un applicazione android già esistente sul dispositivo ma con firma diversa, si deve disinstallare l'applicazione e reinstallarla.

Firma release > firma debug

☒ Da fare: provare 3/5 permission

# NodeJS - 18/10/2023

mercoledì 18 ottobre 2023 10:17

**NodeJS** (linguaggio di programmazione: JavaScript lato server) è un framework che viene utilizzato con AngularJS, ReactJS e MongoDB: serve per creare dei **servizi** (es: DNS, protocollo UDP porta 53).

Un servizio è un programma che esegue delle istruzioni (es: inviare la data corrente).

Il server recupera la chiave primaria e memorizza il record all'interno di una tabella.

Linguaggio di programmazione lato server (es: PHP): gira sul server, creano la pagina web (solo e unicamente HTML, CSS e JS) e viene inviata al client. Essa viene aperta dall'utente tramite il browser, che interpreta la pagina come HTML, CSS e JS.

Differenza tra libreria e framework → **libreria** si importa, il **framework** è un ambiente.

Cos'è **NPM**? E' un portale che contiene dei moduli e componenti già fatti.

Per creare un certificato: **Let's Encrypt**

Il client per comunicare con il server DEVE usare il **browser**.

L'**header della risposta** del pacchetto dice al browser come comportarsi.

L'**header della richiesta** contiene informazioni, come la lingua.

**http** nasce come protocollo senza stato, ovvero senza memoria. Non salva le informazioni. E' **stateless**.

Il browser può salvare la pagina web nella **cache**, in modo da non dover ricaricare ogni volta la pagina. Se però il campo **length** cambia, e quindi vuol dire che è cambiato qualcosa, la pagina viene ricaricata.

Come si crea un **Server Web** (il codice viene *interpretato dal server* e non dal browser).

**Request:** comunicazione tra **CLIENT** e **SERVER**

**Response:** comunicazione tra **SERVER** e **CLIENT**

---

Per passare un **parametro** al **server**, nel client bisogna effettuare la **fetch**.

Il **form** si usa quando il client deve inviare dei parametri al server.

Il tag **form** serve per inviare dei **parametri** al server.

---

**Dispatcher** → serve a navigare tra le pagine del sito tramite uno switch. Gestisce più url.

---

**CommonJS** e **ES modules**, servono per esportare e importare i moduli.

**CommonJS** → standard di nodejs, usa **require**, usa **export**.

**ES modules** → standard di javascript, usa **import**, usa **export const**

**import** è asincrono e importa solo quello che serve grazie alla destrutturazione, **require** è sincrono (blocca il programma) e importa TUTTO il modulo.

Tutte le funzioni devono avere l'**export**.

File **package.json** → contiene tutti i moduli

Per crearlo → **npm init**

Per usare il modulo ES modules aggiungere il campo :

**"type": "module",**

Modificare il campo in questo modo:

**"main": "server.js",**

Con **ES modules** si usa **export const** e non **exports**.

Per startare → **npm start** o **nodemon server.js** o **node server.js**

---

**async** → la funzione **RIMANE SINCRONA**, non trasforma la funzione in asincrona. Permette di usare **await** e di avere il **then**.

**await** → la funzione deve essere dichiarata con **async**, ritorna una promise e permette di utilizzare **then**, **catch**, e **finally**.

Solo dopo che ha eseguito la funzione eseguirà le operazioni successive.

---

Il modulo **Express.js** è un dispatcher, viene utilizzato per controllare gli url.

Non è nativo, deve essere scaricato con → **npm init** e poi **npm install express**

Differenza tra **GET** e **POST** → GET (limiti caratteri, parametri in chiaro nell'url), POST (non c'è un limite, i parametri sono nel payload)

In **POST**:

**Header** (come si deve comportare la pagina) e **Payload** (parametri in chiaro, devono essere cifrati con il certificato)

Si possono recuperare i parametri con lo split (stringa), oppure possono essere visti come json e per recuperarli basta mettere `.nomeparametro`

```
app.use(express.json());
app.use(
  express.urlencoded({
    extended: true,
  })
);
```

Modulo **cors**: serve a gestire i parametri inviati al server.

```
app.use(cors());
app.option("*", cors());
```

---

**Template engine** → permettono di inserire nelle pagine dati aggiornati.

Il template è **server**.

Non è client perché si collega al db, carica i dati nell'html creando una pagina live e lo spedisce al client. Il client visualizza la pagina nel browser facendo Visualizza sorgente pagina, non ispeziona perché ispeziona ti fa vedere la pagina che si modifica in base alle cose che succedono in quel momento.

Estensione del file **.ejs**, oppure **.jeld** o **.twig**.

```
app.set('views', './views'); //Dove si trovano i template, ovvero in views
app.set('view engine', 'ejs'); //Cosa vogliamo usare, ovvero un engine di tipo ejs
```

```
//Funzione render, si trova all'interno di res. Prendo i dati, invia i dati al template, costruisce la pagina e invia la
pagina al client
//Render vs sendFile --> sendFile prende la pagina così com'è e invia al client mentre render inserisce anche il parametro
res.render("index",{
  //Object literal, chiave valore
  variabile:dati
});
```

```
<!-- Codice EJS -->
<!-- E' codice interpretato dal server, il client interpreta solo HTML CSS e JS -->
BENVENUTO <%=variabile%>
```

---

A cosa servono i **socket**? Servono per far comunicare tra loro due host. Il socket non ha bisogno del browser (no http/https). I web socket hanno bisogno del browser (si http/https).

Esempio: bluetooth usa i socket.

Che protocollo usare? TCP o UDP. TCP è orientato alla connessione (prima di trasmettere il pacchetto deve instaurare una connessione con il server). UDP non c'è il server ed è connectionless.

Altra differenza tra TCP e UDP: se si perde un pacchetto, TCP lo ritrasmette, UDP no. Esempio in streaming se si perdono dei pacchetti non è un problema e quindi si usa UDP, in chat si quindi si usa TCP.

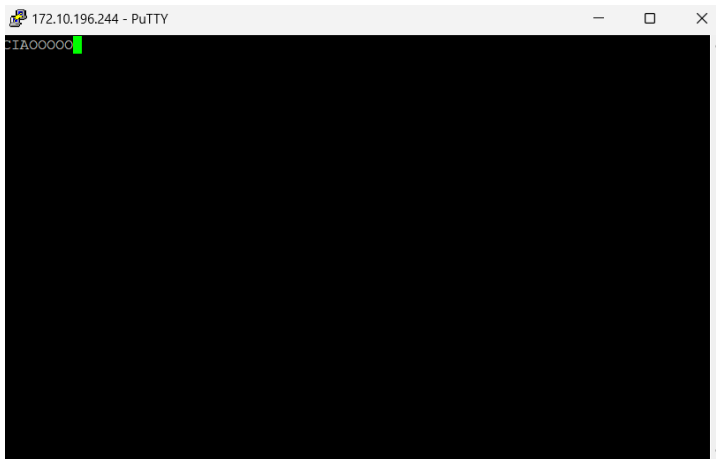
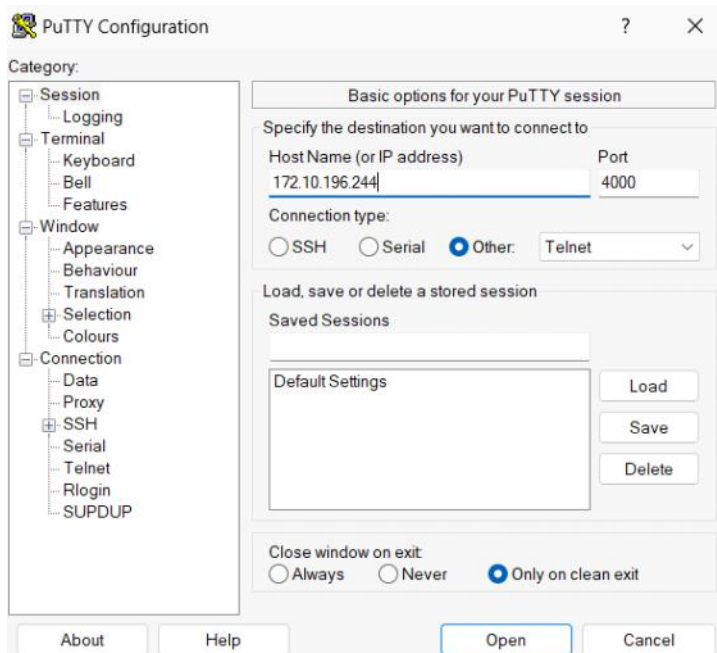
0-1023: Porte Note

1024-49151: Porte Registrate

49152-65535: Porte non registrate (possono essere usate)

Per usare i socket --> `npm install net` (possibilità di creare server, client, utilizzare il protocollo UDP)

Download Putty: è un client e si può usare SSH o Telnet (noi usiamo Telnet)



<https://www.ionos.it/digitalguide/siti-web/programmazione-del-sito-web/che-cose-websocket/>

A cosa servono i **websocket**? Giochi online, aste (ebay), chat, live sport, social. Ad esempio, se arriva un messaggio su instagram, non devo aggiornare la pagina perché resta sempre in ascolto.

Se non si vogliono usare i websocket, si usa il tag refresh, che ogni tot secondi aggiorna automaticamente la pagina.

E' un protocollo, è nativo.

Useremo Socket.IO

Il client deve poter utilizzare il websocket.

Perché si usano? Il client contatta il server, il server accetta la connessione (orientato alla connessione), si instaura la connessione, non c'è più bisogno di aggiornare la pagina.

npm install socket.io --> Socket.io non ha la funzione listen, quindi deve essere usato con http o express.

NON SI USA IL FORM, NO GET E POST.

La connessione è full-duplex.

Socket.io:

Join crea una stanza

Broadcast lo invia a tutti

# Arduino - 24/01/2024

mercoledì 24 gennaio 2024 10:15

Analogico: tutti i numeri (0-255)

Digitale: 0 o 1

Solo l'R4 ha il wifi (modulo wifi).

In arduino ci sono due funzioni: setup e loop.

Grazie alla libreria WiFi3 è possibile collegare l'arduino al wifi.

```
#include <WiFi3.h>
```

`WiFi.begin("WIFI-STUDENTI", "WIFI-STUDENTI");` → connette l'arduino al WiFi, inserendo nome e password.

Codice:

```
// #include "secrets.h"
#include <WiFi3.h>
// uint8_t *mac;
WiFiServer server(3000);
void setup() {
    pinMode(11, OUTPUT);
    Serial.begin(115200);
    server.begin();
    delay(1000);
    Serial.println("Simple WiFi");
    WiFi.begin("WIFI-STUDENTI", "WIFI-STUDENTI");
    while (WiFi.status() != WL_CONNECTED) {
        delay(500);
        Serial.print(".");
    }
    /*
    Serial.print("MAC: ");
    WiFi.macAddress(mac);
    Serial.print(mac[0]);
    Serial.print(mac[1]);
    Serial.print(mac[2]);
    Serial.print(mac[3]);
    Serial.print(mac[4]);
    Serial.print(mac[5]);
    */
    Serial.print("IP: ");
    Serial.println(WiFi.localIP());
}
void loop() {
    // listen for incoming clients
    WiFiClient client = server.available();
    //se il client invia i dati (quindi >0) salvo il carattere nella variabile carattere
    if (client.available() > 0) {
        // read the bytes incoming from the client:
        Serial.print(client.remoteIP());
        char carattere = client.read();
        //invio al client il carattere letto dal client
        server.write(carattere);
        //stampo nel serial monitor il carattere inviato dal client
        Serial.write(carattere);
        //se il carattere inviato dal client è uguale a E chiudo la connessione con il server
        if(carattere=='E'){
```

```

        client.stop();
    }
    //se il carattere inviato dal client è uguale a A accendo il LED
    if(carattere=='A'){
        digitalWrite(11,HIGH);
        Serial.print("ACCESO");
    }
    //se il carattere inviato dal client è uguale a S spengo il LED
    if(carattere=='S'){
        digitalWrite(11,LOW);
        Serial.print("SPENTO");
    }
}
// close the connection:
// client.stop();
/*
Socket CLIENT
WiFiClient client;
    if (!client.connect("172.10.196.202", 3000)) {
        Serial.println("Connection to host failed");
        delay(1000);
        return;
    }
    Serial.println("Connected to server successful!");
    client.print("Hello from ESP32!");
    Serial.println("Disconnecting...");
    client.stop();
    delay(10000);
*/
}

```



# MongoDB - 27/03/2024

mercoledì 27 marzo 2024 10:22

Appunti sul quaderno (anno scorso)  
Appunti nella cartella --> Galeasso\_MongoDB

E' un database NOSQL, ovvero non ci sono relazioni  
I dati vengono salvati in BSON.  
Lato server.

Installare il modulo mongodb nella cartella del progetto --> npm i mongodb

Costruttore --> metodo che non ha il return, viene richiamato una sola volta, stesso nome della classe. Serve per costruire la classe.  
Metodo statico --> possiamo usare il metodo statico senza istanziarlo.  
Attributo statico --> dal momento che si dichiara un attributo statico, è un attributo condiviso tra le istanze.  
Funzione ha le tonde, campo o attributo no.

insertMany --> array di json  
insertOne --> un json

deleteMany  
deleteOne

updateMany --> aggiorna tutti  
updateOne --> aggiorna il primo che trova

find  
findOne



**RITORNANO UNA PROMISE**

.sort({nome:1}) --> crescente  
.sort({nome:-1}) --> decrescente

.project({nome:0}) --> fa vedere tutti tranne il campo nome  
.project({nome:1}) --> fa vedere solo nome

//update --> il primo parametro è il documento (il filtro), il secondo parametro è sempre un documento, e serve per modificare.  
//L'ultimo parametro è sempre un documento --> upsert. E' un boolean (true, false). Se è false, se non c'è il documento cercato non fa nulla. Se è true, se non c'è il documento cercato lo crea.  
miaCollection.updateMany({cognome:"Rossi"},{\$set:{nome:"Blu"}},{upsert:true})

# Valutazione docenti - 10/04/2024

lunedì 22 gennaio 2024 10:49

## Progetto **Valutazione Docenti**

Su Filezilla:

- File
- Nuovo sito
- Parametri:
  - 192.168.10.240
  - Porta: 2999
  - Quintal
  - tuttaSalute24

# React Native - 10/04/2024

mercoledì 24 aprile 2024 10:17

Riguardare appunti Android Studio precedenti.

React Native --> è un framework. Crea delle app native.

Angular e Cordova creano web app: si creano degli apk, ma contengono un browser.

Con React invece si creano degli apk (o IPA) nativi per il sistema operativo.

Sdk, jdk, emulatore, variabile d'ambiente (dice dove si trova l'sdk).

Comando per creare un nuovo progetto --> `npx create-react-native-app nome_progetto`

Cartelle:

- Android: si trova `AndroidManifest.xml`, si trovano le permission.

Come si fa partire un progetto --> `npm run android`

ADB --> serve per gestire gli input del telefono, ecc...

Per creare un componente --> `rnfes`

Nel return non si può usare codice html o css, ma i componenti di react native (View, Text, ecc...)

View è un div.

`JAVA_HOME` --> senza bin

Modal --> componente della libreria react native, è come un alert

Status bar --> barra in alto

Quando si inserisce un'immagine, l'altezza e la larghezza dell'immagine è a 0 pixel. Bisogna modificarla per vederla.

# Lezione 1 - 23/11/2023

giovedì 23 novembre 2023 14:39

**SAP** è un gestionale

**ABAP** è il linguaggio

Ci sono **3** figure: **sistemista**, **funzionale** e **programmatore**.

DDL: create

DML: delete, update (modifica), insert.

SELECT (output)

FROM (tabelle)

WHERE (input + collegamenti)

DISTINCT() → evita le ripetizioni

BETWEEN() → tra

IN() → cerca gli elementi in un sottoinsieme

---

```
ACQ(ID_A,DATA,N_PEZZI,PAG,EVASO)
PROD(ID_P,DESCR,NUM,P2,SOGLIA,ID_A)
```

```
SELECT PROD.DESCR
FROM PROD
WHERE PROD.NUM < PROD.SOGLIA
```

```
SELECT(PROD.NUM*PROD.P2) AS "VALORE"
FROM PROD
WHERE PROD.DESCR
```

3. Visualizzare tutte le date di acquisto di un certo prodotto

```
SELECT ACQ.DATA
FROM ACQ,PROD
WHERE PROD.ID_P = [...]
      AND ACQ.ID_A = PROD.ID_A
```

4. Visualizzare i prodotti di cui sono stati acquistati dai 20 ai 40 pezzi

```
SELECT PROD.* //Visualizza tutta la riga
FROM ACQ,PROD
WHERE ACQ.N_PEZZI > 20 AND N_PEZZI < 40
WHERE ACQ.N_PEZZI BETWEEN 20 AND 40
```

5. Prodotti venduti nel trimestre del 2023

```
SELECT PROD.*
FROM PROD,ACQ
WHERE ACQ.DATA BETWEEN #01/01/2023# AND #31/03/2023#
```

---

```
CLIENTI(ID_C,NOME,CITTA)
```

1. Visualizza le città di tutti i clienti

```
SELECT DISTINCT(CLIENTI.CITTA)
FROM CLIENTI
```

2. Clienti che abitano a Torino, Milano, Genova

```
SELECT CLIENTI
WHERE CLIENTI.CITTA="Torino" OR CLIENTI.CITTA="Milano" OR CLIENTI.CITTA="Genova"
WHERE CLIENTI.CITTA IN("Milano","Torino","Genova")
```

---

COUNT - SUM - AVG - MAX - MIN → valori numerici (tranne il COUNT)  
GROUP BY non fa parte della query (la query finisce dopo il WHERE)

```
ACQ(ID_A,DATA,N_PEZZI,PAG,EVASO)
PROD(ID_P,DESCR,NUM,P2,SOGLIA,ID_A)
```

```
SELECT MAX (PROD.P2) AS "MASSIMO"
FROM PROD
```

```
SELECT MIN (ACQ.N_PEZZI) AS "MINIMO"
FROM ACQ
WHERE ACQ.EVASO = [NO]
ORDER BY()
GROUP BY(ACQ.EVASO)
```

# Lezione 2 - 30/11/2023

giovedì 30 novembre 2023 14:55

Su SAP:

Togliere il risparmio energetico (tutte e tre le spunte)

Terminale → dir → cd abaptrial → cd inst → dir → sudo ./install.sh → password: utente → ctrl c → yes → Password#000 → invio

---

ORDER BY()

GROUP BY()

Stanno fuori dalla query!

SUM COUNT AVG MAX MIN → Funzioni di aggregazione

Non si raccoglie mai per chiave primaria, prezzo, voto, anno, ecc...

Si raccoglie per cognome, chiave secondaria...

1. Contare tutti i prodotti che hanno prezzo tra 10 E 20

```
SELECT COUNT(ART.ID_A) AS "..."
```

```
FROM ART
```

```
WHERE ART.PR BETWEEN 10 AND 20
```

2.

```
SELECT COUNT(ART.ID_A) AS "..."
```

```
FROM ART
```

```
WHERE ART.N_PEZZI < ART.SOGLIA AND ART.ID_F == ""
```

GROUP BY (ART.ID\_F)

3. Calcolare il totale dei prezzi presenti di un certo fornitore

```
SELECT SUM (ART.N_PEZZI) AS "..."
```

```
FROM ART
```

```
WHERE ART.ID_F==""
```

GROUP BY(ART.ID\_F)

4.

```
SELECT ART.DESC
```

```
FROM ART
```

```
WHERE ART.PR = SELECT.MAX(ART.PR) AS ""
```

```
FROM ART
```

5. Fammi vedere la descrizione degli articoli il cui prezzo è maggiore della media dei prezzi

```
SELECT ART.DESC
```

```
FROM ART
```

```
WHERE ART.PR > SELECT AVG(ART.PR)
```

```
FROM ART
```

---

I programmatori modificano il gestionale (es: aggiungono un button, ecc...)

SAP → gestionale

ABAP → linguaggio di programmazione

Il server può essere chiuso e mandato in esecuzione.

Dal momento che si spegne il computer il servizio viene stoppato, per fare ripartire il servizio → su -c "startsap all" -l npladm

Password: Password#000

Per collegarsi al server in alto a sinistra → Nuovo → Connessione → Continuare → SAP LOGIN NPL 00 127.0.0.1 → Eseguire Log On → si apre una finestra

DEVELOPER

Download

Transizioni → codice, permettono di fare qualsiasi cosa

license → mostra le licenze

Per uscire dalla transizione → pulsante exit, oppure /n la transizione che vuoi aprire

se38 → serve per creare record

Lente → permette di vedere i report e stamparli

Tutti i record devono iniziare con la Z o con la Y

Tutto quello che facciamo deve sempre essere attivato

Se11 → permette di creare tabelle o caricare dati nella tabella

# Lezione 3 - 7/12/2023

giovedì 7 dicembre 2023 14:44

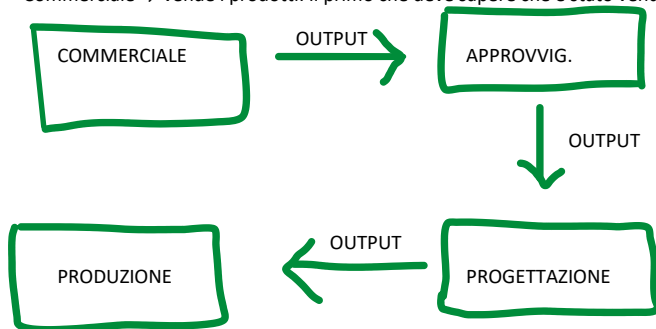
## GESTIONE DELLA QUALITA'

**PROCESSO:** insieme di attività legate ad uno obiettivo che hanno un input e un output; l'output di un processo aziendale è sempre l'input per il processo successivo.

**AZIENDA:**

- **PROCESSI PRIMARI:** **approvvigionamento** (influiscono direttamente sulla qualità del prodotto), **commerciale** (basso costo = bassa qualità), **progettazione** (progettazione base = bassa qualità), **produzione / erogazione del servizio**.

Commerciale → vende i prodotti. Il primo che deve sapere che è stato venduto un trattore è l'approvvigionamento.



- **PROCESSI SECONDARI (o DI SUPPORTO):** sono trasversali ai processi primari che non coinvolgono un singolo processo ma tutti i processi primari. **DIREZIONALE**, **PROCESSO DELLE RISORSE UMANE** (esistono d'appertutto), **LOGISTICO** (comprende tutto il flusso dal commerciale alla produzione), **ATTREZZATURE**, **MAGAZZINO**.

CED (parte informatica, fondamentale per l'azienda).

Vantaggio del SAP: prende le informazioni, le elabora ed esce fuori con dei consigli. Facilita la vita di ogni singolo modulo (ovvero ogni singolo processo).

Ogni processo è composto da **procedure**, **istruzioni** e **moduli** al fine di garantire sempre la rintracciabilità dell'informazione.

**Procedure:** insieme di regole utili alla gestione di un processo.

**Istruzione:** è una parte della procedura.

Procedura dell'APPROVVIGIONAMENTO:

Procedura a 5 persone diverse (riserve)

Richiesta scritta

Risposta

Riesame dell'offerta

Scelta del fornitore

Modello di ordine

Richiede che il fornitore mandi la richiesta d'ordine

SAP ti dice ad esempio a che fornitore interpellare.

**Istruzione:**

Come si riesce ad adempiere alla procedura, come svolgere la procedura.

**Moduli:** Evidenza/informazione documentata a supporto delle istruzioni e delle procedure. Output di quello che abbiamo fatto (esempio: conferma d'ordine).

Nei processi è fondamentale che abbia dei **controlli**, ogni processo deve essere **monitorato**. Avviene attraverso gli **indicatori**.

**Indicatori:** sono dei valori che delineano l'andamento di un obiettivo da raggiungere andando quindi a valutare la bontà di un processo o di una parte di esso.

Esempi di indicatori: tempistiche (tempi dei fornitori, se arriva in ritardo, licenzia), analisi dei prezzi (benchmarking, concorrenza).

Ogni indicatore deve essere **S.M.A.R.T:** Specific Misurable Achievable Rilevant Timebased

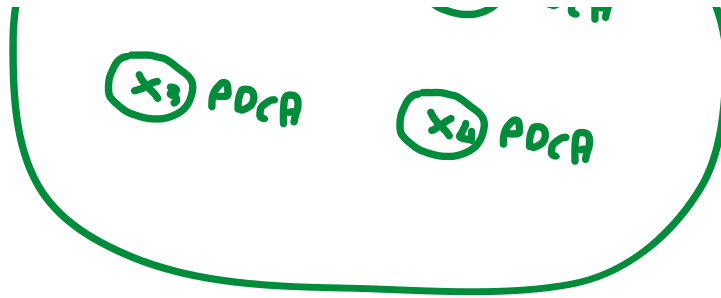
Non esiste un indicatore senza un tempo di monitoraggio.

**Pdca (CICLO DI DEMING) → PLAN DO CHECK ACT**



$x_i =$  PROCESSO





#### Pdca PROCESSI PRIMARI:

- **Pdca PRODUZIONE:**  
ACT: 100 pezzi → 20 sbagliati. SAP il giorno dopo dirà di farne altri venti.
- **Pdca COMMERCIALE:**  
PLAN: Pianificazione  
DO: Fare la vendita (commessa)  
CHECK: Verifica  
ACT:
- **Pdca APPROVVIGIONAMENTO:**  
PLAN: In base alla distinta base, magazzino e logistica si pianifica  
DO: Procedure istruzioni e modelli  
CHECK: Controllo con la conferma d'ordine  
ACT: Se non viene confermato (non conformità), abbassa il rating del fornitore

#### • VALUTAZIONE FORNITORI:

Fornitore	CRITICO	NO CRITICO
	Trasportatore di Pezzo cromato rarissimo	Fornitore di penne

- **CHECK e ACT Approvvigionamento:**  
Se **CRITICO**: visitare il fornitore, controllo pezzi uno a uno.  
Se **NON CRITICO**: pezzi conformi in tempo, controllo delle certificazioni.
- **Pdca PROGETTAZIONE:**  
PLAN: Verificare le direttive (leggi),  
DO: Calcoli strutturali, disegni formano il fascicolo tecnico. Dopo → riesame della progettazione.  
CHECK: C'è e non c'è. Se c'è → ad esempio → fare il prototipo  
ACT: Se il prodotto è andato male, si rifà. Economia di scala → più ne produco, più vinco.

#### Pdca PROCESSI DI SUPPORTO:

- ...

**ISO 9001:2015** → norma internazionale per gestire al meglio la qualità aziendale, ovvero la qualità di ogni singolo di processo.

**OUT-SOURCING**: metto il mio nome ma è l'altro che fa per me. Se l'altro fa un casino sono cavoli miei.

# Lezione 4 - Assente

giovedì 21 dicembre 2023 14:46

# Lezione 5 - 21/12/2023

giovedì 21 dicembre 2023 14:47

Transition: → Composta da 4 caratteri, il primo rappresenta il modulo, la seconda rappresenta documento, ultimo numero indica che puoi creare qualcosa (1 creare 2 modificare 4 elimina).

se11 → permette di creare tabelle o caricare dati nella tabella. Permette di vedere, creare, modificare database

/nse16 → se sei nella transition ad esempio se11 e vuoi andare nella se16, ti sovrascrive la scheda con la nuova transizione.

/ose16 → apre una nuova scheda

Creazione di un database.

Il database è unico, si chiama HANA.

Dictionary object → permette di creare domini, data element, struttura, database e delle view (relazione tra le tabelle).

Si crea dominio, data element e poi tabella.

DOMINI:

Domain → deve iniziare con z o con y + il nome del dominio. Esempio zmi dominio e poi si fa su create.

z\* puoi vedere tutti i domini che hai creato e che puoi modificare.

Successivamente mettere la short description.

I domini si creano per

DOMINIO VS DATA ELEMENT:

Dominio rappresenta la struttura del campo (dimensione e tipo di dato), il data element è la descrizione del campo.

Se modifichi il dominio tutti i dati verranno aggiornati.

Quando si crea il DB: tabella di tipo A (dati anagrafici che vengono modificati raramente e dati di transizione che vengono modificati frequentemente), C (vengono usate per personalizzare l'applicazione), L (usata per memorizzare dati temporanei), G (SAP in cui può aggiungere dati ma non modificarli), E (dati che servono al sistema), S (solo SAP può modificarle la tabella), W (tabella con dati di amministrazione di sistema).

A

Prima opzione: non puoi modificare

Seconda: puoi fare tutto

Terza: non puoi fare niente

APPL0 (dati anagrafici, aggiornati poco)

APPL1 (dati della transazione, aggiornamenti frequenti)

APPL2 (dati che servono all'organizzazione, alla ditta, e per personalizzare)

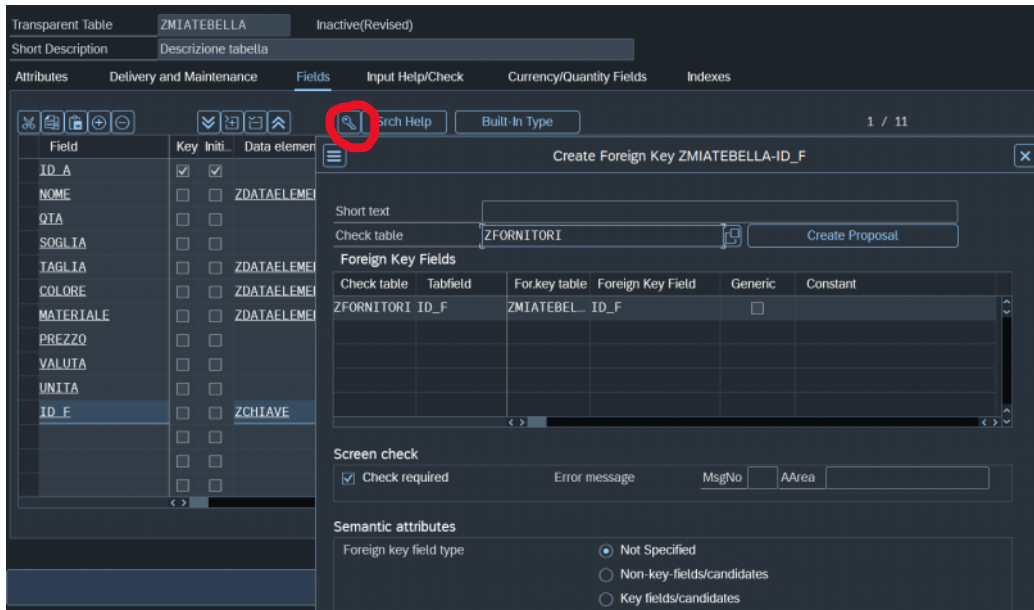
Utilities, table content, create entries

# Lezione 6 - 18/01/2024

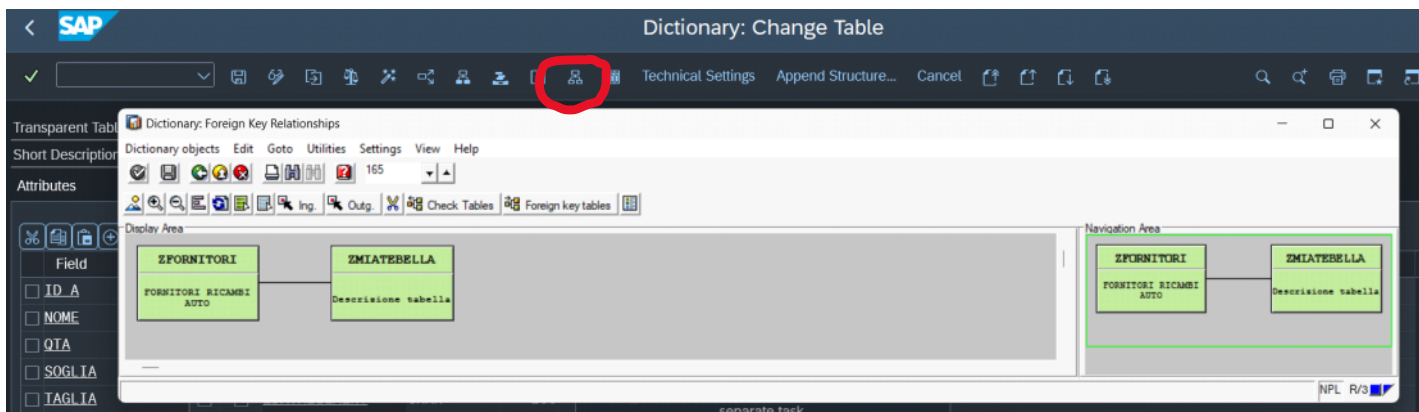
giovedì 18 gennaio 2024 14:40

Il data-element deve sempre essere associato alla chiave primaria.

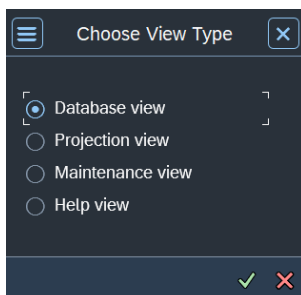
Per la **chiave esterna** (creare la relazione):



Per vedere le **relazione**:



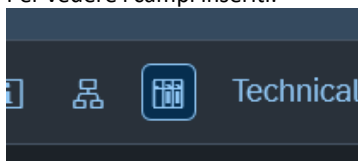
Le **viste** servono per manipolare le tabelle che hanno la relazione (estrapolare i dati):



- 1) Vedere ma non modificare (SELECT)
- 2) Proiezioni: estrapola solo i campi che interessano
- 3) Lettura e scrittura
- 4) Manualletto

Per creare (inserire i campi):  
Utilities --> Table content --> Create Entries

Per vedere i campi inseriti:



## Lezione 7 - 25/01/2024

giovedì 25 gennaio 2024 14:40

PER ATTIVARE --> BACCHETTA MAGICA E POI LOCAL OBJECT

sm30 --> serve per un'interfaccia grafica, per caricare i dati nella vista. BISOGNA DARE I PERMESSI PER POTERRE UTILIZZARE LA VISTA. --> Maintain.

Per dare i permessi --> utilities --> table maintenance generator

The screenshot shows the SAP Table Maintenance Generator (TMO) interface for table ZCARICAFORNITORI. The interface is divided into several sections:

- Table/View:** ZCARICAFORNITORI
- Technical Dialog Details:**
  - Authorization Group: &nc&
  - Authorization object: S\_TABU\_DIS
  - Function group: ZMIOGRUPPO
  - Package: (empty)
- Maintenance Screens:**
  - Maintenance type: ☒ two step
  - Maint. Screen No. 1: Overview screen
  - Maint. Screen No. 2: Single Screen
- Dialog Data Transport Details:**
  - Recording routine: ☒ no, or user, recording routine
  - Compare Flag: Automatically Adjustable
  - Note button: i Note

Per salvare --> salva --> local object

## PROGRAMMAZIONE

Possiamo creare due tipi di programmi: eseguibili (possiamo creare dei report o dei pull di moduli) e non eseguibili (classi, interfacce, pull di subroutine, include di moduli).

Eseguibile:

Report --> estrapolo delle informazioni dei database da salvare da qualche parte

Per creare i report --> se38 --> Create

The screenshot shows the SAP ABAP: Program Attributes ZPROVA Change dialog. The dialog is divided into several sections:

- Title:** ZPROVA
- Original language:** EN
- Created:** DEVELOPER 08.02.2024
- Last Changed:** DEVELOPER 08.02.2024
- Status:** New (Revised)
- Attributes:**
  - Type: Executable program
  - Status: Unclassified
  - Authorization Group: (empty)
  - Application: (empty)
  - LDB name: (empty)
  - Selection screen: (empty)
  - ABAP Language Version: Standard ABAP (Unicode)
  - ☒ Fixed point arithmetic
  - ☐ Editor lock
  - ☐ Start using variant

At the bottom of the dialog, there are buttons for Save, Edit, Find, and other actions.

NB:

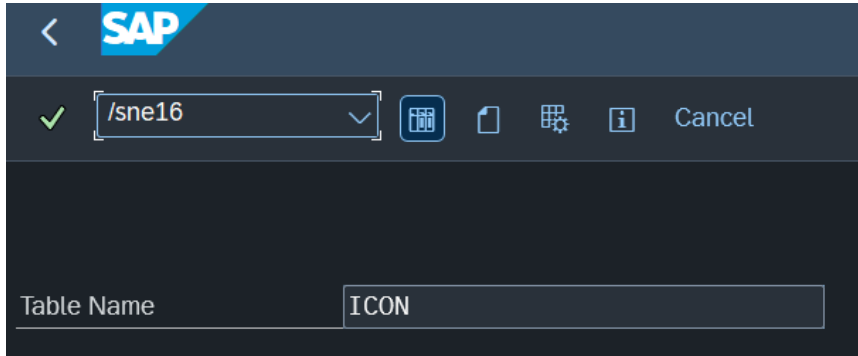
Procedura non ritorna nulla, la funzione ritorna qualcosa al chiamante.

### Linguaggio ABAP

Per stampare:



Per vedere le icone:



Codice ABAP:

```
*&-----*
*& Report ZTEST
*&-----*
*&
*&-----*
REPORT ZTEST.

*Commento su tutta la riga
"Commento a metà della riga

*Dichiarazione di una variabile
data prova(5) type c.

*Dichiarazione di più variabili
data: nome(10) type c,
      cognome type string,
      eta type i,
      TOTALE TYPE p decimals 2.

nome = 'Mario'.
eta = 5 + 1.

*Creazione di una struttura (object literal)
DATA: BEGIN OF ARTICOLO,
      NOME TYPE STRING,
      PREZZO TYPE P DECIMALS 2,
END OF ARTICOLO.

ARTICOLO-nome = 'MOUSE'.
ARTICOLO-prezzo = 54.

*Stampa nel report
write /'IIS Denina'.
write /'Ragioneria ITIS'.
ULINE.
SKIP.
write /10 ARTICOLO-NOME.
write 30 ARTICOLO-PREZZO && '€'.
write /10 'MONITOR'.
write 30 '150.00€'.
write /10 'CASE'.
write 30 '100.00€'.
SKIP.
ULINE /10(30).
SKIP.
*NEW-PAGE.
write /10 'TOTALE'.

TOTALE = 45 + 150 + 100.
WRITE: 30 '@3Z@', TOTALE && '€' COLOR 5.

SKIP.
ULINE.
SKIP.
```

```

*Richiamo di una procedura
PERFORM SOMMA USING 4 5.

SKIP.
ULINE.
SKIP.

*Richiamo di una procedura
PERFORM MOLTIPLICAZIONE USING 10 9.
PERFORM MOLTIPLICAZIONE USING 10 10.

SKIP.
ULINE.
SKIP.

*Richiamo di una procedura
PERFORM CONTROLLO_NUM USING 2 1.

*Creazione di una procedura
FORM SOMMA USING VALUE(A)
                        VALUE(B) .
  DATA TOT_SOMMA TYPE I.
  TOT_SOMMA = A + B.

  WRITE TOT_SOMMA.
ENDFORM.

*Creazione di una procedura
FORM MOLTIPLICAZIONE USING VALUE(A)
                        VALUE(B) .
  DATA TOT_MOLT TYPE I.
  TOT_MOLT = A * B.
  IF TOT_MOLT >= 100.
    WRITE TOT_MOLT COLOR 5.
  ELSE .
    WRITE TOT_MOLT COLOR 6.
  ENDIF.
ENDFORM.

*Creazione di una procedura
FORM CONTROLLO_NUM USING VALUE(A)
                        VALUE(B) .

  IF A > B.
    WRITE A COLOR 5.
  ELSE .
    WRITE B COLOR 6.
  ENDIF.
ENDFORM.

```



# Lezione 8 - 8/02/2024

giovedì 8 febbraio 2024 14:53

sy --> variabili di sistema

TABELLE INTERNE (array): con o senza intestazione (più usate)

se11 --> spfli --> display

Per passare un parametro alla procedura si usa using, per passare una tabella si usa tables

# Lezione 9 - 15/02/2024

giovedì 15 febbraio 2024 15:57

```
*&-----*
*& Report ZVOLI
*&-----*
*&
*&-----*
REPORT ZVOLI.

data: itab_voli type table of spfli,
      wa_voli type spfli.

select * from spfli into table itab_voli where cityfrom = 'FRANKFURT'.

loop at itab_voli into wa_voli.
  write: / wa_voli-cityfrom, wa_voli-distance.
ENDLOOP.

write: / .
uline.
write: / .

***** TABELLA VOLI *****
data: itab_flight type table of sflight,
      wa_flight type sflight.

select * from sflight into table itab_flight where currency = 'EUR'.

loop at itab_flight into wa_flight.
  IF wa_flight-price < 500.
    write: / wa_flight-price color 6.
  ELSE .
    write: / wa_flight-price color 5.
  ENDIF.
ENDLOOP.
```

# Lezione 10 - 20/02/2024

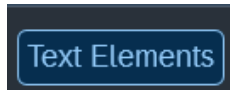
martedì 20 febbraio 2024 14:42

Parametri.

WorkArea --> 1 riga

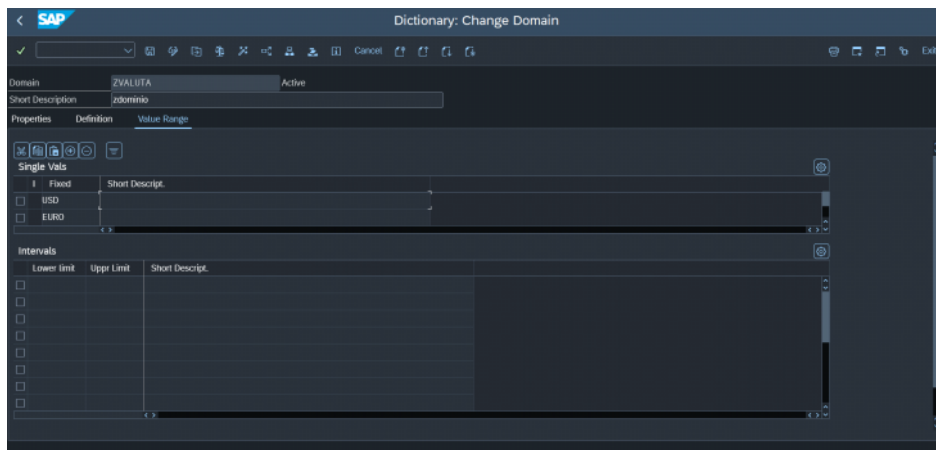
Tabella --> tante righe

Text Elements --> Selection Texts: serve per modificare il nome del parametro.



Quando il checkbox/radio è selezionato il parametro contiene il valore impostato, diversamente non contiene nulla.

Creazione dominio per lavorare su codice se38, con i valori che vogliamo noi:



Codice:

```
*&-----*
*& Report ZPARAMETRI
*&-----*
*&
*&-----*
REPORT zparametri.
```

```
PARAMETERS codice TYPE string.
PARAMETERS value TYPE zvalutetable-valuta AS LISTBOX VISIBLE LENGTH 40.
SELECTION-SCREEN PUSHBUTTON /10(20) TEXT-001 USER-COMMAND pippo.
```

```
*PARAMETERS USD AS CHECKBOX.
*PARAMETERS EURO AS CHECKBOX.
*PARAMETERS JPY AS CHECKBOX.
```

```
*PARAMETERS USD RADIOBUTTON GROUP VALU.
*PARAMETERS EURO RADIOBUTTON GROUP VALU.
*PARAMETERS JPY RADIOBUTTON GROUP VALU.
```

```
DATA: itab_voli TYPE TABLE OF sflight,
      wa_voli   TYPE sflight.
```

```
*Evento: serve per controllare casella di testo piena o vuota. VIENE ESEGUITO PRIMA DI C
REAREIL REPORT
*Popup: E (errore, puoi anche chiudere il report), I (informazione), success, warning.
```

```

AT SELECTION-SCREEN.

IF sy-ucomm = 'PIPPO'.
    codice = ''.
    value = ''.
ENDIF.

IF value IS INITIAL.
    MESSAGE 'SLEZIONA LA VALUTA!' TYPE 'E'.
ENDIF.

*Serve creare il report
START-OF-SELECTION.

    IF codice IS INITIAL.
        CASE value.
            WHEN 'USD'.
                SELECT * FROM sflight INTO TABLE itab_voli WHERE currency = 'USD'.
            WHEN 'EUR'.
                SELECT * FROM sflight INTO TABLE itab_voli WHERE currency = 'EUR'.
            WHEN 'JPY'.
                SELECT * FROM sflight INTO TABLE itab_voli WHERE currency = 'JPY'.
            WHEN OTHERS.
                ENDCASE.
        ELSE.
            CASE value.
                WHEN 'USD'.
                    SELECT * FROM sflight INTO TABLE itab_voli WHERE currency = 'USD' AND carrid = c
odice.
                WHEN 'EUR'.
                    SELECT * FROM sflight INTO TABLE itab_voli WHERE currency = 'EUR' AND carrid = c
odice.
                WHEN 'JPY'.
                    SELECT * FROM sflight INTO TABLE itab_voli WHERE currency = 'JPY' AND carrid = c
odice.
                WHEN OTHERS.
                    ENDCASE.
            ENDIF.
        LOOP AT itab_voli INTO wa_voli.
            WRITE: / wa_voli-carrid, wa_voli-price, wa_voli-currency.
        ENDLOOP.
    END-OF-SELECTION.

```

# Lezione 11 - 29/02/2024

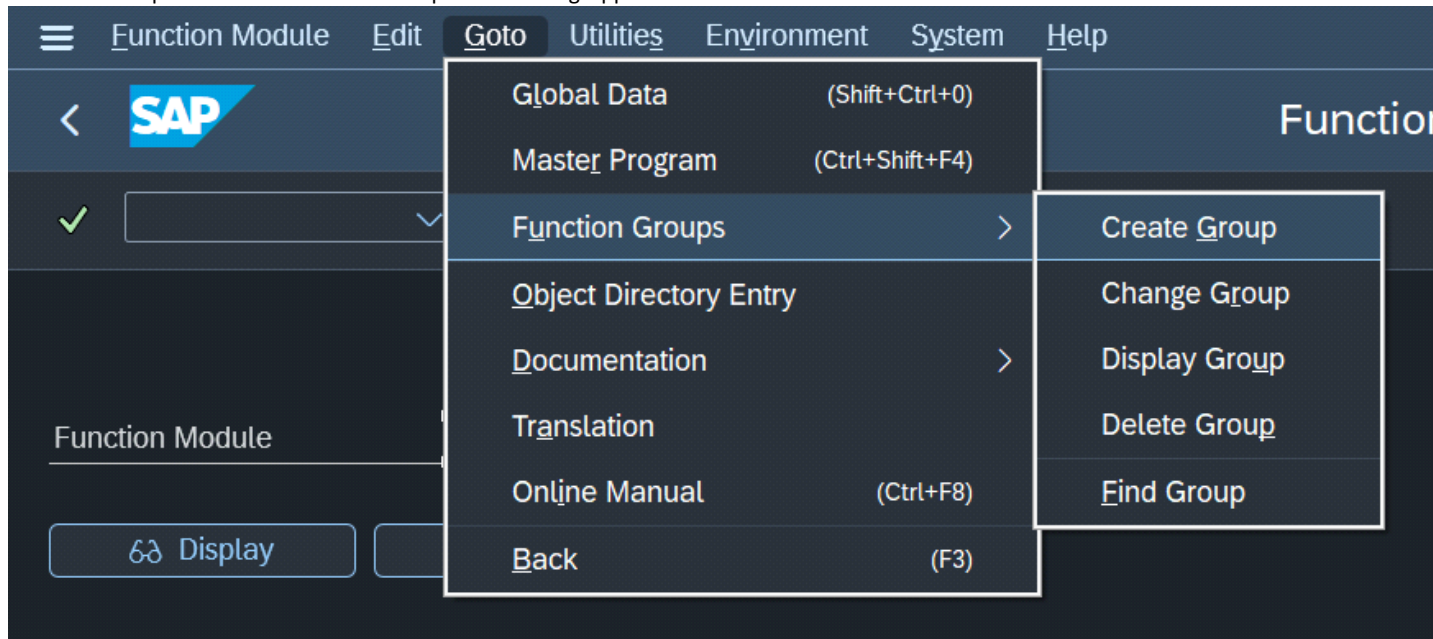
giovedì 29 febbraio 2024 15:02

1. Quanti aerei arrivano a francoforte (5)
2. Dato in input il volo di partenza, stampare dove va
3. Tutti i voli che hanno un tempo di volo tra le 10 e 15 ore (BETWEEN)

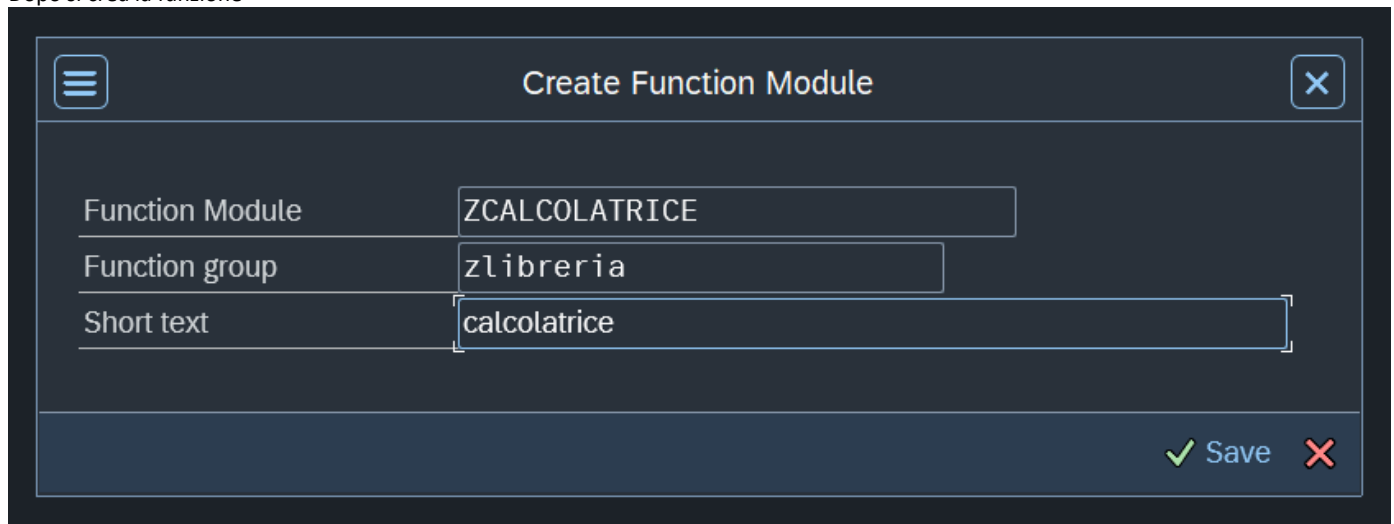
## Lezione 12 - 07/03/2024

giovedì 7 marzo 2024 14:37

Se37 --> serve per creare le funzioni. Serve per creare un gruppo di funzioni



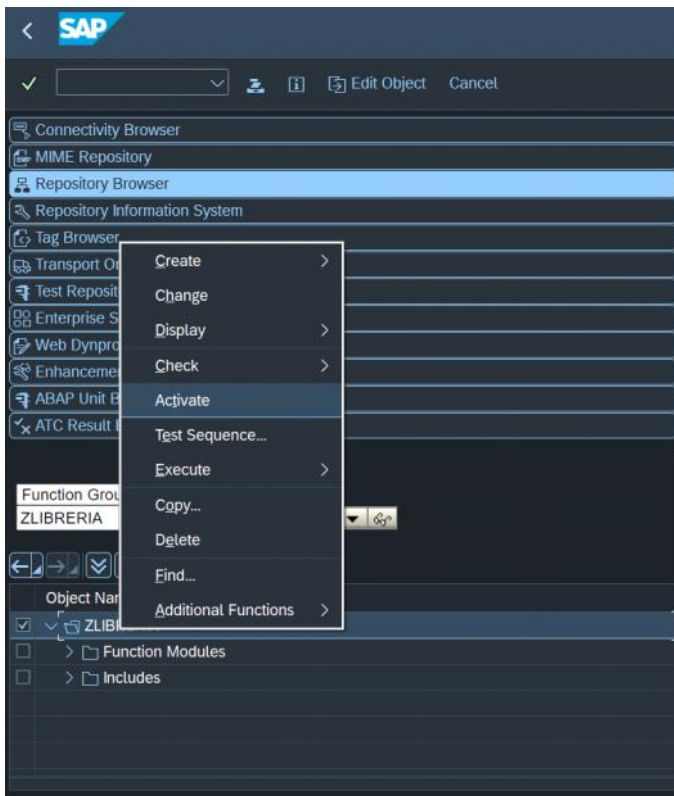
Dopo si crea la funzione



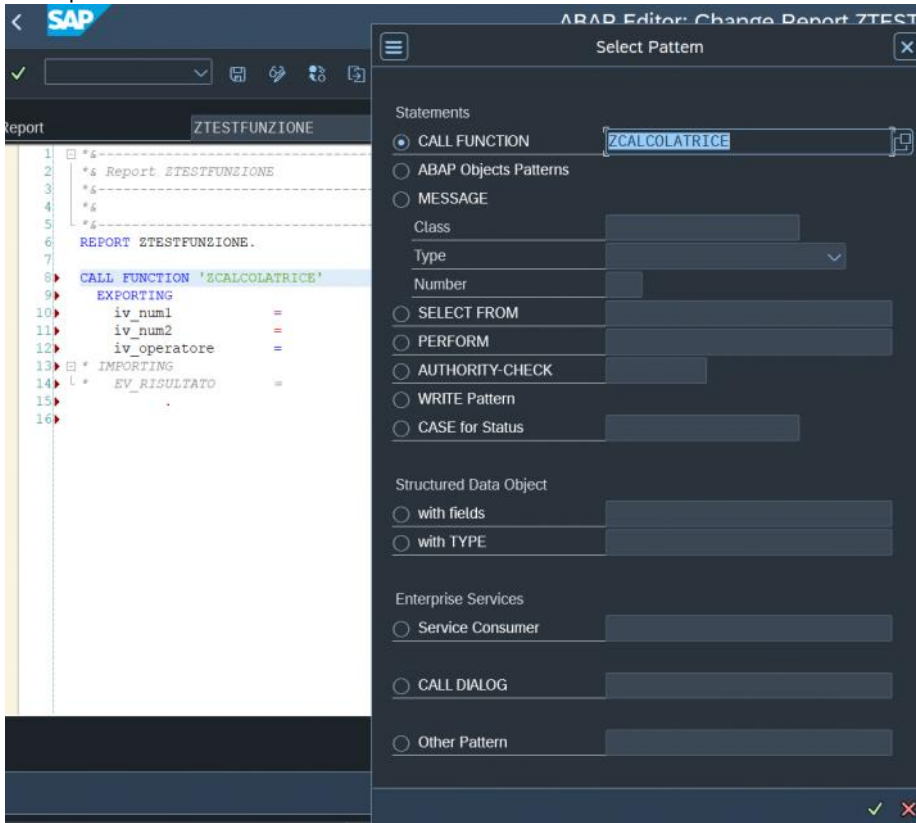
La variabile import solitamente ha un prefisso: IV (import variabile). Esempio IV\_num1.

La variabile export solitamente ha un prefisso: EV (import variabile). Esempio EV\_risultato.

Se80 --> serve esplorare il progetto e per attivare il gruppo



Per importare una funzione --> CTRL - FN - F6



!!! Togliere i commentida IMPORTIN e EV\_RISULTATO

Codice:

```
*&-----*
*& Report ZPROVAFUNZIONE
*&-----*
*&
*&-----*
REPORT ZPROVAFUNZIONE.
```

```
data: itab_voli type table of spfli,
      wa_voli type spfli.
```

```

data: nomecampo type string,
      valorecampo type string.

select * from spfli into table itab_voli.

loop at itab_voli into wa_voli.
  write:/40 wa_voli-cityfrom color 4, 80 wa_voli-cityto color 5, wa_voli-connid color 1.
endloop.

at line-selection.

*Ritorna il nome del campo e il secondo parametro è il contenuto.
get cursor field nomecampo value valorecampo.

*Serve per creare una nuova finestra
WINDOW STARTING AT 1 1 ending at 100 100.

IF nomecampo = 'WA_VOLI-CONNID'.
  select * from spfli into table itab_voli where connid = valorecampo.
  LOOP AT itab_voli into wa_voli.
    write:/ wa_voli-cityfrom color 4, wa_voli-cityto color 5.
  ENDLOOP.
ENDIF.

```

Report nuovo, cliccando su cityfrom bisogna avere come risultato tutte le destinazioni, con ora di partenze e arrivo.



# Lezione 13 - 21/03/2024

giovedì 21 marzo 2024 14:42

RIPASSO