

Data Mart "la Vigilanza" (VGL)

Questo esercizio prevede la realizzazione di un ETL in plsql per il caricamento di un Data Mart in ambito sanitario (la Sacra Famiglia di milano <http://www.sacrafamiglia.org>).

Il progetto nasce dall'esigenza, da parte del cliente di monitorare le ore di lavoro del proprio personale suddivise per ogni unità e per mansione (o ruolo). Questo monitoraggio nasce sia per un interesse dell'Istituto stesso che per conto dell'ASL la quale effettua controlli periodici sul servizio erogato.

Dal nostro Data Mart vengono prodotte analisi quali ad esempio:

- Quante **ore** hanno erogato i Medici nell'Unità San Giovanni nel mese di Febbraio 2015?
- Quanto **personale** ha lavorato per ogni mese nelle unità accreditate come RSD?

Quindi i fatti o misure della fact nostro DM sono ore e "teste" (conteggi di persone) e non "soldi".

Introduzione

Il progettino del nostro data warehouse/data mart è strutturato sui classici due livelli:

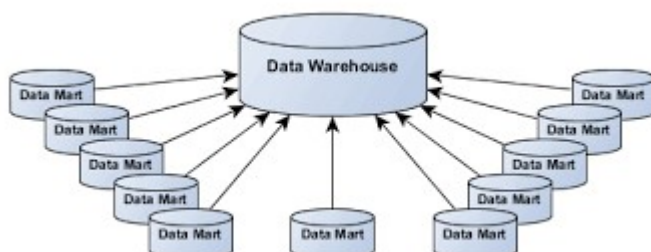
1° livello = "staging area"

2° livello = "data warehouse", nel nostro piccolo esempio, si parla direttamente di data mart (per togliere ogni dubbio, per dwh si intende, in genere (poi dipende dalla realtà del cliente) l'insieme "logico" dei "data mart" che sono "piccoli data warehouse" specializzati per "ambito di business".

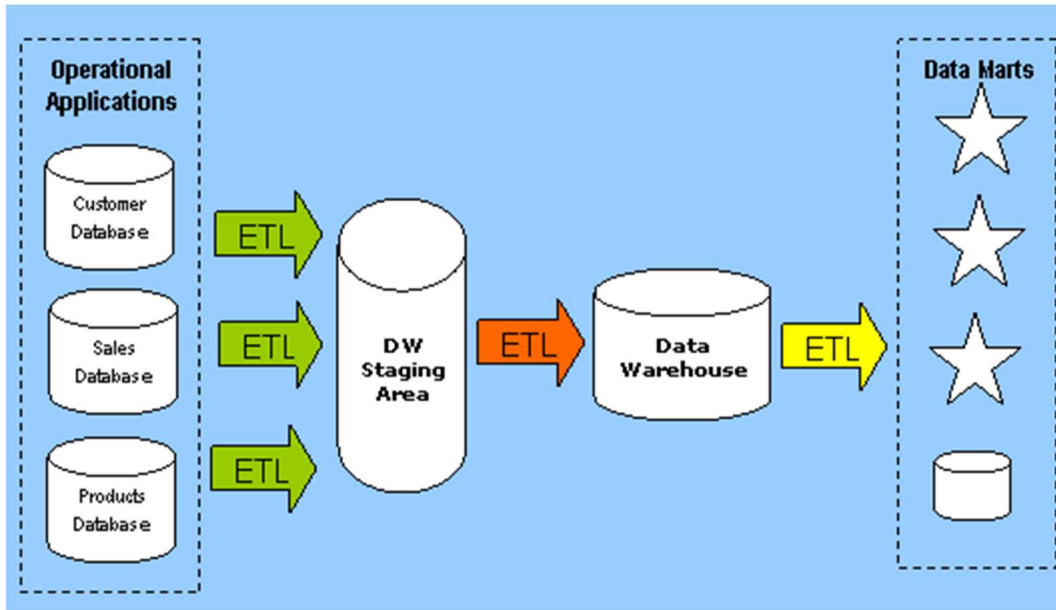
ad esempio, in ambito assicurativo, potreste trovare (e magari lavorare) sui questi data mart:

- DM "polizze"
- DM "preventivi"
- DM "life" (incentrato sulle polizze vita)
- DM "reclami"
- DM "danni"

L'unione "logica" di questi DM rappresenta, magari, per quel cliente il "Data Warehouse" come riportato qui in figura:



Mentre l'immagine qui sotto tipica del mondo "data warehouse", può essere fuorviante, in quanto tende a rappresentare un altro "modo di vedere" i Data Mart come "figli tematici" della grande mamma "Data Warehouse".



Entrambe le "visioni" sono valide e, ripeto, dipendono dalla realtà del cliente anche se quella da me riportata come esempio sulle assicurazioni è quella che più frequentemente troverete al lavoro.

Per la creazione del Data Mart quindi sono stati implementati due livelli:

- **VGLSA** = Staging Area

il primo livello del [DWH](#), denominato universalmente "Staging Area " (area di appoggio, di stage) serve tipicamente ad effettuare le validazioni sui dati che si andranno poi ad inserire all'interno del [Data Mart](#) (2° livello, a [stella](#)) (nel nostro esercizio, sul "VGLDM").

Nel nostro esercizio, vengono effettuate le seguenti validazioni:

- sui data type presenti nei flussi (tipicamente numeri e date)

Altri tipici controlli del processo di [ETL](#) del 1° livello sono:

- campi chiave non nulli
- doppioni
- campi "fuori dai range" (es: una età negativa)
- aggiunta di colonne "calcolate" (es: si aggiunge il campo calcolato "codice fiscale" a partire dai dati base dell'anagrafica)

Eventuali anomalie riscontrate in fase di ETL del 1° livello vengono gestite in due modi:

- Arresto completamente il processo di etl
- Continuo il caricamento dell'etl

di solito questa scelta viene fatta con il cliente che, in base al proprio "business" decide se portare avanti il caricamento e, nel caso, POI gestire eventuali scarti o se, in caso di scarti, fermare completamente il caricamento per gestirli subito e poi ri-processare il flusso "sporco". Quando scrivo

"gestirli subito", voglio dire che tipicamente, si va a capire il "perché" vi siano dati sporchi nel flusso... quindi si va a parlare con coloro che producono i nostri flussi di caricamento che tipicamente provengono dai database dei cosiddetti sistemi alimentanti o [data source](#) del DWH (tipicamente i ["gestionali o ERP"](#) presenti all'interno della realtà del cliente).

Per entrambe le modalità si va SEMPRE a tracciare eventuali dati sporchi in apposite tabelle di scarti. ATTENZIONE a non confondere gli "scarti" con altre tabelle "tecniche" che contengono invece i "LOG" dei caricamenti. In quest'ultime infatti vengono memorizzate informazioni tipo:

- data inizio caricamento
- flusso caricato
- procedura che lo ha elaborato
- esito del caricamento (es: KO o OK, e nel caso di "KO", volendo anche il numero di SCARTI).

Nel nostro esercizio sono presenti sia tabelle di SCARTI che tabelle di LOG su entrambi i due livelli implementati (le vedremo dopo).

Qui trovate un valido link relativo alla gestione degli scarti:

<http://blog.aditi.com/data/etl-how-to-handle-bad-data>

- VGLDM = Data Mart

il 2° livello del nostro esercizio, il "Data Mart", rappresenta "la [stella](#)" finale sulla quale poi verranno effettuate le analisi richieste dal cliente. Apro una parentesi, per sottolineare che il "risultato finale" della realizzazione di un "data mart" è quello di rendere disponibile al cliente un db specializzato su una certa tematica sul quale effettuare ANALISI. Ne parlo al plurale perché le analisi implementate sono tipicamente "tante" e dipendono soprattutto dalle "dimensioni" presenti all'interno della struttura a stella che "spiegano" i fatti / misure presenti nella fact table. Queste Analisi vengono poi spesso "accorpate" in Dashboard al fine di produrre "pagine sintetiche" con quante più possibili informazioni. Ecco un esempio di Dashboard:



I principali tool di mercato specializzati nella cosiddetta ["Business Intelligence"](#) sono:

"la BI di oracle oppure OBI" [Oracle Business Intelligence Suite](#) (EE=enterprise edition)

"BO" [SAP Business Objects](#)

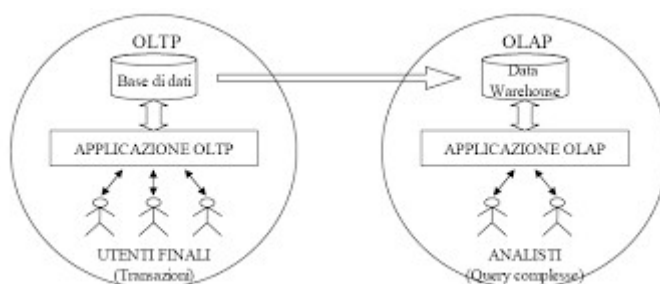
"Click View" [QLIK](#)

Da non confondere, mi raccomando, la BI (Business Intelligence) con i "REPORTS", quando si parla di reporting tipicamente ci si riferisce a stampe operative (bolle, fatture, buste paga, scontrini...) e sono assolutamente dei "must" sui software gestionali/erp. Ovviamente questo non vuol dire che con i tool di BI non si possa produrre reports... ma sarebbe come andare con la "ferrari in prima"... anche le fasce di prezzo dei tool di BI sono molto più elevate dei tool di reporting . Insomma sono "cose differenti".

A tal scopo ricordo che i sistemi alimentanti/gestionali sono sistemi orientati alle transazioni (OLTP) mentre noi, sul dwh, siamo sistemi orientati alle Analisi (OLAP). La differenza è sostanziale come potete vedere dalla tabellina sotto riportata.

	OLTP	OLAP
users	clerk, IT professional	knowledge worker
function	day to day operations	decision support
DB design	application-oriented	subject-oriented
data	current, up-to-date detailed, flat relational isolated	historical, summarized, multidimensional integrated, consolidated
usage	repetitive	ad-hoc
access	read/write index/hash on prim. key	lots of scans
unit of work	short, simple transaction	complex query
# records accessed	tens	millions
#users	thousands	hundreds
DB size	100MB-GB	100GB-TB
metric	transaction throughput	query throughput, response

Proprio per questo anche i modelli delle base di dati dei sistemi gestionali sono strutturati in modo completamente differente dal disegno del db dei nostri dwh o data mart.



Insomma olap e oltp sono come il giorno e la notte...proprio perché gli scopi sono praticamente opposti l'un l'altro.

Quanto pensate ai gestionali (i nostri "sistemi alimentanti") dovete pensare a maschere tipo questa:



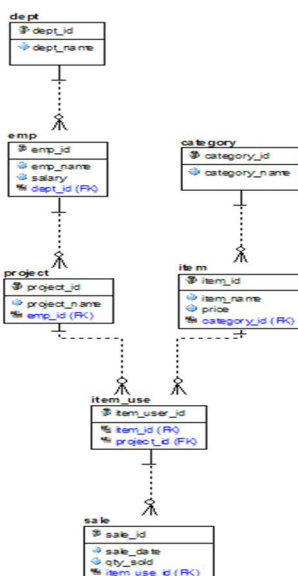
con DB in 3FN e con le "famose" STAMPE OPERATIVE "a corredo". Dove gli utenti finali sono spesso "operativi" (tipo "le segretarie" degli uffici amministrazione). Con questi gestionali, ricordate, che si fanno soprattutto insert, update e delete "puntali" (su singoli record) e, ovviamente stampe operative.

Mentre sul dwh si fanno solo SELECT MASSIVE su grossi moli di dati e gli utenti sono molti di meno e tipicamente "decisionali" (es: la direzione dei crediti di una banca). Quindi il risultato sono Analisi e Dashboard.

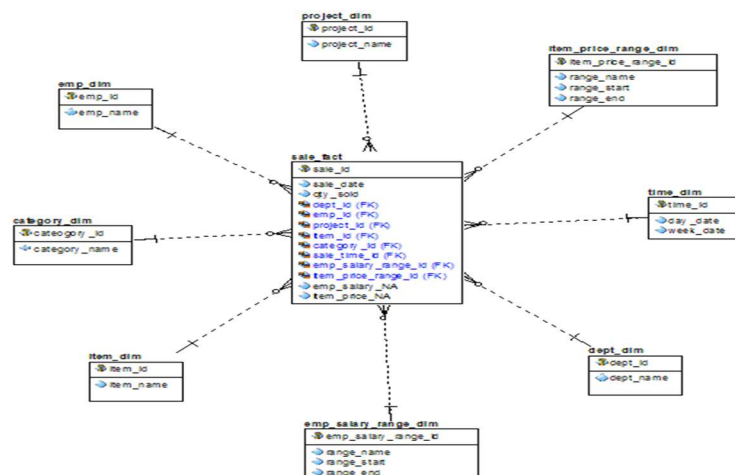
Infatti per rispondere a queste differenze quindi i "classici" disegni dei db gestionali sono, come insegnano a scuola, "ben fatti o [normalizzati](#)", cioè si dice in 3FN (terza forma normale) mentre i nostri del dwh sono appunto STAR SCHEMA e quindi "[denormalizzati](#)" proprio per dar maggior "forza" alle interrogazioni massive (analisi).

La seguente immagine dovrebbe trasmettere bene questo importantissimo concetto:

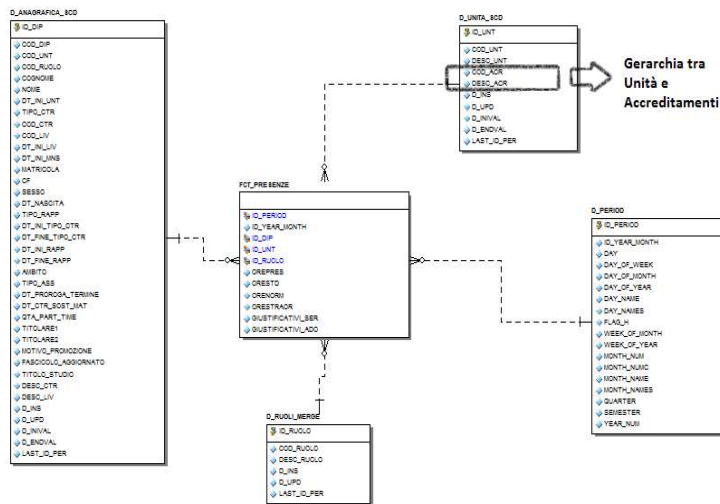
3FN



STAR SCHEMA



Tornando all'esercizio, quindi sul secondo livello, dopo la validazione dei dati del 1° livello, effettueremo in estrema sintesi la denormalizzazione dei dati per caricare la nostra "stellina" che nello specifico sarà questa:



Ne approfitto per ribadire un paio di concetti sullo star schema:

Abbiamo due tipi di "tabelle" **Dimensioni** e **Fatti** e tali strutture solo relazionate da constraint logici (e non fisici, per motivi di performance) la cui coerenza è gestita dal processo di ETL stesso.

Dimensioni:

sono "banalissime" anagrafiche" (eventualmente denormalizzate) che hanno come scopo quello di "spiegare" i fatti di interesse. Non a caso, banalizzando all'estremo, abbiamo sempre:

Chi?, Cosa?, Dove? e Quando? quindi, banalmente;

Cliente - Prodotto - Negozio - Calendario

Considerazioni:

- La dimensione temporale è una "tabella STATICA" caricata ad "inizio progetto" con banalissime procedure o "fogli excel" (non serve un flusso di alimentazione per dirci che domani sarà oggi +1...) Trovate un esercizio di come popolare la D_TIMES sul sito stage:

http://www.advancia.it/stage/oracle/docs/caricamentoD_PERIODO.pdf



D PERIODO.pdf

- Sulle dimensioni viene creata una nuova colonna (come prima) che tipicamente tutti chiamano "ID". Tale colonna è numerica e fa da **chiave** della tabella (ricordo che non si implementano i Constraint fisicamente) e va a "sostituire" (soprattutto per problemi di performance) le "chiavi" presenti nei flussi e quindi nei gestionali (che "noi" chiamiamo "CODICI"). Vi ricordo che una chiave del gestionale, che essendo un oltp non ha problemi particolari di performance, potrebbe essere composta anche da più colonne (es: Cognome, Nome) mentre noi qui abbiamo appunto problemi di essere molto performanti. Questi ID si chiamano [SURROGATE KEY](#) proprio perché surrogano (fanno le veci) delle chiavi "originali" del gestionale (i "codici"). In Oracle, noi gestiamo gli ID con semplici sequence.
- Dimensioni senza storicizzazione (tipicamente in insert/update o "[merge](#)")
- Dimensioni con storicizzazione ([SCD](#)) (la TYPE 2)

Dal primo livello ci arriva un record già presente da caricare in una dimensione (es: un prodotto), se la dimensione non gestisce la storicizzazione, allora faremo una update PERDENDO i valori presenti prima dell'operazione di aggiornamento (in pratica, sovrascriviamo i dati) altrimenti, si storicizza: si chiude il vecchio record e se ne apre uno nuovo. In questo modo eventuali analisi nel "passato" non verranno potenzialmente rovinate dai nuovi valori caricati in quanto punteranno alla "foto vecchia" del record. Pensate se ad esempio la "coca cola" cambiasse "tipologia"... passando da "bevanda" a "casalingo"... che impatto si potrebbe avere su una analisi di confronto di venduto sugli ultimi tre anni per tipologia...dall'oggi al domani cambierebbero tutti i valori...quindi la "s" di "slowly" indica che si storicizza tipicamente solo per attributi IMPORTANTI (ai fini delle analisi) e che, essendo importanti, cambiano "raramente"... In fase di analisi del progetto si concorda con il cliente se una dimensione sia o meno da gestire in scd e su quali attributi "attivare" questa storicizzazione. Ad esempio potremmo ipotizzare che se cambia il "colore" del prodotto NON si storicizza mentre se cambia, come prima, la tipologia, allora sì.

(*) Attenzione che con le SCD, la chiave della tabella a questo punto è solamente l'ID e non il codice dell'alimentante in quanto possono esservi più "foto/versioni" del record. Mentre sulle dimensioni senza l'scd anche la chiave "codice" resta chiave in quanto non ho mai più copie dello stesso record (cmq la chiave da considerare è sempre l'ID che andremo a mettere nella fact table).

NB

Tipicamente si storicizza al variare delle colonne "che fanno parte" della gerarchia e che al 99% erano tabelle nel gestionale (poi denormalizzate)...

Provo a spiegare:

se nel gestionale erano presenti queste tre tabelle (ognuna ad esempio 10 attributi)

Famiglia_Prodotto - Tipologia_Prodotto - Prodotto con valori quali ad esempio:

Food - Bevande - CocaCola

sulla Dimensione Prodotto avremmo tutti e 30 gli attributi denormalizzati (più l'id per primo)...

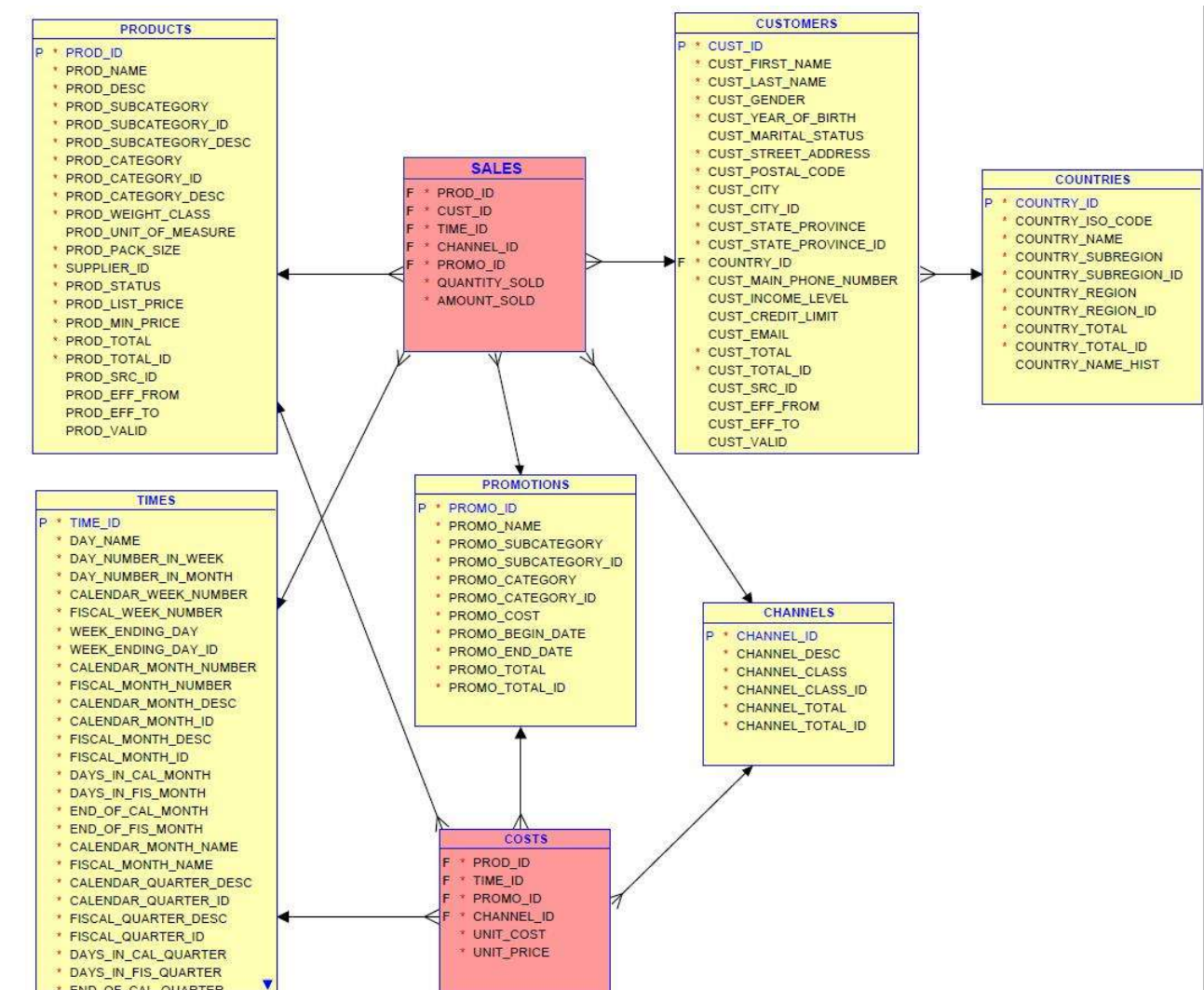
Allora, al 99%, se si dovesse fare la storicizzazione molto probabilmente questa scatterebbe, tra le 30 colonne al variare o della tipologia o della famiglia. Questo per "deduzione" dovuta che sul sistema gestionale queste info erano molto importanti tali da averle gestite in opportune tabelle. Mentre se cambiassero i valori sugli altri attributi (quali il colore) allora andremmo in "update" semplice.

Fact Table(s):

sono le tabelle "strette e lunghe" (potete pensare agli scontrini del supermercato) e contengono per definizione:

tutti gli ID delle dimensioni + i fatti o misure del data mart (*quindi SOLO NUMERI*)

Spesso esiste la corrispondenza 1 DB = 1 Fact ma non è sempre vero... ad esempio nello schema SH di Oracle (provate a vederlo) abbiamo due fatti..la sales e la orders.



da notare che come "TIME ID" nello schema SH è stato messo un campo di tipo DATE ma non si fa quasi mai....piuttosto è comodo gestire il time-id come numero parlante...ad esempio YYYYMMGG (20160302) questo per avere un numero "parlante" già sui fatti...solo per comodità. Ma mai un date!

le fact sono spesso molto grosse come numero di record e solitamente sono [partizionate](#) / sottopartizionate.

Ovviamente le fact table ragionano sempre in INSERT e si caricano per ULTIME proprio perché dentro ci vanno gli ID delle Dimensioni.

La chiave di partizionamento viene scelta solitamente in base alle "interrogazioni più frequenti" che verranno fatte sulla fact. A mò di esempio in "banca" facciamo così:

```
PARTITION BY RANGE (ID_DPER)
SUBPARTITION BY LIST (CODABIDRV)
```

quindi partizioniamo sul periodo e sotto partizioniamo sul codice abi.

Apro una parentesi sui "flussi" di alimentazione.

Flussi (formati)

possono essere:

- con separatore (tipicamente ";" e quindi "csv")

UltraEdit - [C:\Users\lugo\Desktop\Vigilanza_Anagrafica_2015_10.csv]

File Edit Search Insert Project View Format Column Macro Scripting Advanced Window Help

01s_DDL.sql Vigilanza_Anagrafica_2015_10.csv x

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1	Data Fine Periodo estrazione	Matericola (cod_dip)	Cognome	Nome	Cod Centro di Conto	Data Inizio CDC	Tipo Contratto	Codice Contratto	Codice Livello	Data Inizio Livello						
2	2015-10-31	000001	SANTORCHI	MARCELLO	MARIA	12141	2015-10-05	STAGIATTI	010000	001	2015-10-05	000580	2015-10-05	000001	STMG	292R
3	2015-10-31	0004479	VALLI	BERNARDINA	MARIA	15445	1972-11-16	CASE DI CURA PRIVATE	PERS.NON MEDICO	0012	D3	2014-12-01	000652	1972-11-16	0004479	VLLNRS
4	2015-10-31	0004729	CAZZANI	RITA	13300	1974-02-11	CASE DI CURA PRIVATE	PERS.NON MEDICO	0012	C2	1974-02-11	000518	1974-02-11	0004729	CZRT	158S
5	2015-10-31	0004757	CAVALLO	ARIELINA	15450	1974-04-01	CASE DI CURA PRIVATE	PERS.NON MEDICO	0012	D5	2014-11-01	000516	1974-04-01	0004757	CVLDR	4846
6	2015-10-31	0004814	NUNCA	PASSERONI	RAUL	12330	1974-03-11	CASE DI CURA PRIVATE	PERS.NON MEDICO	0012	B1	1974-03-11	000340	1974-03-11	0004814	NMCR
7	2015-10-31	0004852	RUJUI	GAVINIO	15445	1974-10-22	CASE DI CURA PRIVATE	PERS.NON MEDICO	0012	C2	1974-10-22	000190	1974-10-22	0004852	RUJOWNS	6E31A977W
8	2015-10-31	0004862	FRUENTI	MARCELLO	15370	1974-11-26	CASE DI CURA PRIVATE	PERS.NON MEDICO	0012	C2	1974-11-26	000190	1974-11-26	0004862	FRUOMC	52A6
9	2015-10-31	0004907	GATTO	CLARA	30350	1975-03-24	CASE DI CURA PRIVATE	PERS.NON MEDICO	0012	B3	1975-03-24	000500	1975-03-24	0004907	GTACLS	28S02450
10	2015-10-31	0004908	DI GIORGIO	POTTA	15560	1975-03-24	CASE DI CURA PRIVATE	PERS.NON MEDICO	0012	D2	1975-03-24	000614	1975-03-24	0004908	DORPITS	3M69A463W
11	2015-10-31	0005083	MOINO	RITA	14331	1976-02-16	CASE DI CURA PRIVATE	PERS.NON MEDICO	0012	B3	2015-07-01	000176	1976-02-16	0005083	MMRTIS	5C158445
12	2015-10-31	0005224	DEMIRU	TERESA	15320	1976-11-02	CASE DI CURA PRIVATE	PERS.NON MEDICO	0012	B3	1976-11-02	000250	1976-11-02	0005224	IMRT	8557S8B354Z
13	2015-10-31	0005226	RELLI	MARINA	12102	2015-10-01	CASE DI CURA PRIVATE	PERS.NON MEDICO	0012	E2	1976-11-02	000375	1976-11-02	0005226	ELLARNS	5DS2680D
14	2015-10-31	0005294	SCIALA	FRANCESCO	13840	1977-03-01	CASE DI CURA PRIVATE	PERS.NON MEDICO	0012	B1	1977-03-01	000504	1977-03-01	0005294	STUSP	8402B8A77J
15	2015-10-31	0005314	MARONCINI	RITA	15445	1977-04-18	CASE DI CURA PRIVATE	PERS.NON MEDICO	0012	C2	1977-04-18	000190	1977-04-18	0005314	MMNR	158S72133S
16	2015-10-31	0005341	PARADISO	FRANCESCO	15620	1977-05-30	CASE DI CURA PRIVATE	PERS.NON MEDICO	0012	B1	1977-05-30	000504	1977-05-30	0005341	PLDFWC	5B348C51G
17	2015-10-31	0005365	FABRIZI	MARIA CRISTINA	12400	1977-07-11	CASE DI CURA PRIVATE	PERS.NON MEDICO	0012	E1	1977-07-11	000375	1977-07-11	0005365	FBMR	5US812F050
18	2015-10-31	0005382	RIEOLZI	DIEGO	60310	1977-09-12	CASE DI CURA PRIVATE	PERS.NON MEDICO	0012	E2	1977-09-12	000530	1977-09-12	0005382	RLDGT	51904F205Y
19	2015-10-31	0005387	BERNARDI	ILIANA	14250	1977-09-24	CASE DI CURA PRIVATE	PERS.NON MEDICO	0012	C1	1977-09-24	000518	1977-09-24	0005387	BRNTIS	7E180504W
20	2015-10-31	0005396	BISCOSI	SALVATORE	15450	1977-10-05	CASE DI CURA PRIVATE	PERS.NON MEDICO	0012	B3	1977-10-05	000504	1977-10-05	0005396	BSCSV	574615F15Z
21	2015-10-31	0005423	BUGGIERO	ROSALIA	15445	1977-11-14	CASE DI CURA PRIVATE	PERS.NON MEDICO	0012	B1	1977-11-14	000504	1977-11-14	0005423	ROGR	305654G230D
22	2015-10-31	0005483	CFRANZI	CLAUDIO	13850	1978-03-13	CASE DI CURA PRIVATE	PERS.NON MEDICO	0012	B2	1978-03-13	000385	1978-03-13	0005483	CFRC	53277F050
23	2015-10-31	0005513	PAOLINI	DORA MARCELIA	14500	1978-06-19	CASE DI CURA PRIVATE	PERS.NON MEDICO	0012	C1	1978-06-19	000518	1978-06-19	0005513	FLDMNS	57A63C279S
24	2015-10-31	0005550	PALEZZI	ILVIRA	13800	1978-08-16	CASE DI CURA PRIVATE	PERS.NON MEDICO	0012	C2	1978-08-16	000518	1978-08-16	0005550	PVSLVR	39R49C16R
25	2015-10-31	0005553	PALEZZI	STEFANO	15410	1978-09-18	CASE DI CURA PRIVATE	PERS.NON MEDICO	0012	B3	1978-09-18	000504	1978-09-18	0005553	VLMR	5R207DK120
26	2015-10-31	0005587	IMBERTI	LUISA	12400	1978-11-27	CASE DI CURA PRIVATE	PERS.NON MEDICO	0012	D1	2015-05-01	000025	1978-11-27	0005587	MMRL	5US9C4F205D
27	2015-10-31	0005588	CATALANO	PIETRA	11200	1978-11-27	CASE DI CURA PRIVATE	PERS.NON MEDICO	0012	D2	1978-11-27	000588	1978-11-27	0005588	CLTPRT	87L6A745Z
28	2015-10-31	0005589	PACIFICCO	ERILIA	13800	1978-11-28	CASE DI CURA PRIVATE	PERS.NON MEDICO	0012	C2	1978-11-28	000518	1978-11-28	0005589	PCPR	550C62F03F
29	2015-10-31	0005601	SANCINA	FORZLA	14810	1979-01-02	CASE DI CURA PRIVATE	PERS.NON MEDICO	0012	B3	1979-01-02	000504	1979-01-02	0005601	SRCP	257E63B915E
30	2015-10-31	0005617	AMENDUNI	MARIA	12300	1979-02-05	CASE DI CURA PRIVATE	PERS.NON MEDICO	0012	D2	1979-02-05	000025	1979-02-05	0005617	MMDM	5ASD51R445A
31	2015-10-31	0005620	GIORGANO	RICCARDA	15820	1979-02-19	CASE DI CURA PRIVATE	PERS.NON MEDICO	0012	B1	1979-02-19	000504	1979-02-19	0005620	GBML	5R46C145G
32	2015-10-31	0005637	DE CILLIS	GIOVANNI	15450	1979-03-12	CASE DI CURA PRIVATE	PERS.NON MEDICO	0012	B3	1979-03-12	000504	1979-03-12	0005637	DCLO	NN82C13F77W
33	2015-10-31	0005651	LACERENZA	SABINA	30140	1979-04-03	CASE DI CURA PRIVATE	PERS.NON MEDICO	0012	A3	1979-04-03	000345	1979-04-03	0005651	LCRBNS	5C43D643Z

Ln 23, Col 11, CD DOS Mod 26/02/2016 15:51:07 File Size: 707964 INS CAP

- FULL sui flussi delle dimensioni (le anagrafiche)
- DELTA sui flussi delle tabelle transazionali (es: le vendite) e quindi poi le Fact

questo perché ha poco senso stare a caricare il "venduto" di mesi/anni precedenti...basta appunto farsi mandare "gli ultimi giorni" e caricare sempre, salvo errori, il giorno precedente.

Chiudo la parentesi sui "flussi" di alimentazione.

Implementazione

Esempio Report in fase di vigilanza dall'ASL – Accreditemento RSD

VERIFICA DEI REQUISITI GESTIONALI PREVISTI DALLA NORMATIVA PER L'ACCREDITAMENTO

ACCR I ORGANICO	Figura Professionale	Operatori In Organico	Numero Operatori Mese Considerato	Monte ORE mensili	Monte minuti settimanali	Standard				
						CL 1	CL 2	CL 3	CL 4	CL 5
	MEDICO			128	1.748	2.500	2.000	1.600	867	900
	PSICOLOGO				0	N. Ospiti				
	INFERMIERE PROFESSIONALE			724	9.809				36	
	TERAPISTA DELLA RIABILITAZIONE				0	Minuti dovuti				
	ASA-OTA -OSS			2.028	27.476	0	0	0	31.212	0
	OSS				0	Totale		Educatori		12.485
	Educatori				0	Minuti		Assistenza		12.485
	ALTRO:				0	Dovuti		Altro		6.242
	ALTRO:				0	Totale		Educatori		9.809
	ALTRO:				0	Minuti		Assistenza		27.476
	ALTRO:				0	Dati		Altro		1.748
	TOTALE ORE MENSILE DEGLI OPERATORI			2.881		Sovra Standard		Educatori		-2.676
	TOTALE MINUTI SETTIMANALI				39.033	7.821		Assistenza		14.991
	NUMERO OSPITI NEL MESE CONSIDERATO			36				Altro		-4.496

Contesto del progetto

prima di partire provo a sintetizzare alcune macro requisiti.

la "stella" che andremo a popolare è così pensata:

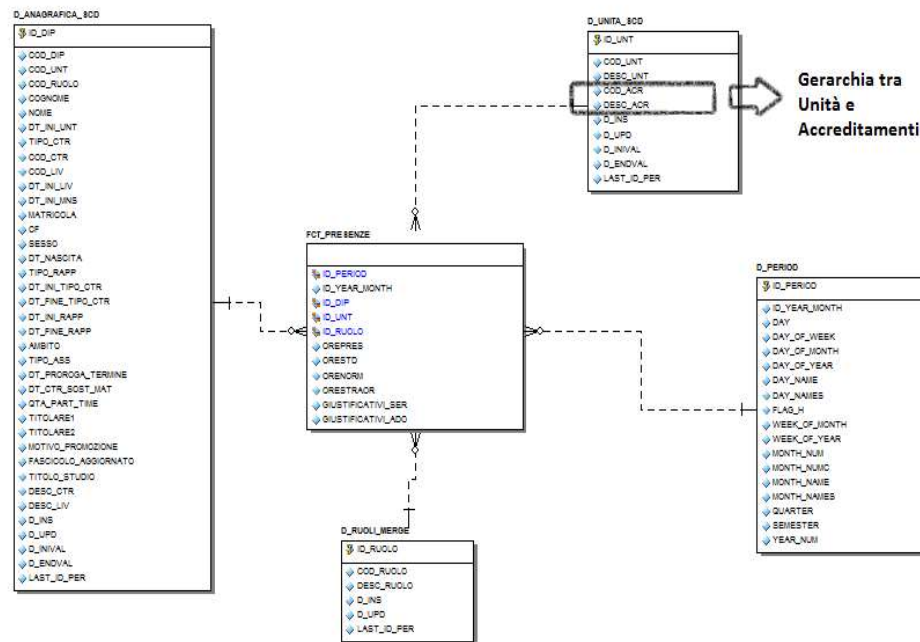
Dimensioni:

- **Personale** Dipendenti (es: mario rossi....)
- **Unità** (es: San Luigi) con gerarchia **Accreditamenti** (es: RSD)
- **Ruoli** (es: Medico) (*avremmo potuto gestirla anche come gerarchia del personale ma ho deciso così solo per non farla "uguale" alle unità*).
- **Calendario**

Fact / Misure

- **"Ore lavorate"** (normali, straordinario e permessi)

la stella sarà infatti questa:

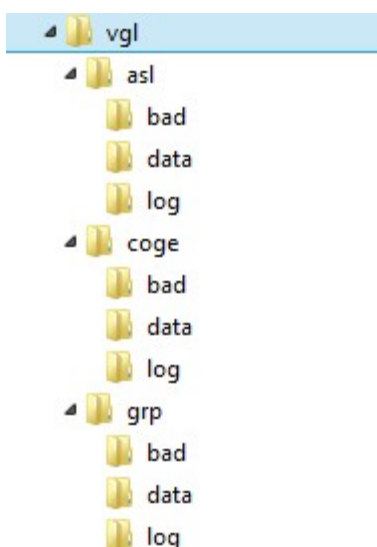


Abbiamo ipotizzato **3 sistemi alimentanti** del nostro data mart

- GRP (gestione rilevamento presenze) :: manda, **anagrafica**, **mansioni**, **presenze** e **unità**
- ASL :: ci manda il flussi degli **accreditiamenti**
- COGE (controllo di gestione) :: ci manda **l'associazione unita / accreditiamenti**

I Flussi Alimentanti

I flussi di prova dovete metterli quindi sotto c: seguendo questa struttura:



dove:

nelle cartelle "data" ci mettiamo i nostri csv di alimentazione

nota:

mentre nelle log ci andranno i log generati dalle "tabelle esterne" e nella bad eventuali record di scarto sempre generati dalle tabelle esterne. I log ed i bad delle tabelle non sono così importanti nel nostro esercizio, ma se volete approfondire (non è necessario), vedete questo link:

https://asktom.oracle.com/pls/asktom/f?p=100:11:0::::P11_QUESTION_ID:363613500346518422

Essendo il processo di ETL un processo ripetitivo, quindi "ogni notte", "ogni settimana", "ogni mese" etc etc ... è fondamentale che all'interno dei flussi vi sia SEMPRE un campo tecnico che dica "A QUANDO SI RIFERISCONO" i record presenti nel flusso. Questo campo è differente dal tempo di invio del flusso o di creazione del file stesso. Attenzione! Ricordatevi sempre che: "oggi posso caricare un flusso di tre anni fa"...!!!

Quindi nei nostri flussi abbiamo deciso di inserire in "TESTATA" il campo ANNOMESE a cui si riferiscono i dati (stiamo ovviamente ipotizzando un ETL con frequenza **MENSILE**).

Esempio in rosso del periodo di riferimento all'interno del file unità accreditamenti (untacr.csv):

```
200909
COD_UNT;COD_ACR
15311;CDD
15312;CDD
15357;CSS
15358;CSS
15360;CDI
15370;CDI
15445;RSA
```

Su altri esercizi fatti durante lo stage sulle "tabelle esterne" o in altri casi avremmo anche potuto inserire il "PERIODO" in colonna in questo modo:

```
PERIODO;COD_UNT;COD_ACR
200909;15311;CDD
200909;15312;CDD
200909;15357;CSS
200909;15358;CSS
200909;15360;CDI
200909;15370;CDI
200909;15445;RSA
```

è chiaro che con questa tecnica si occupa maggior spazio (il campo è ripetuto per ogni riga). Per gestire quindi la necessità di prendere dal file sia il "periodo" che i "dati" abbiamo deciso di fare DUE tabelle esterne su ogni flusso per gestire appunto la doppia necessità.

VGLSA - 1° Livello - Staging Area

Prendiamo ad esempio un flusso e su questo spieghiamo la logica del primo livello.

per creare lo schema basta collegarsi come system e fare copia/incolla ed eseguire lo script **01 CREADB VGLSA.sql** poi vi scollegate da system e vi collegate col nuovo utente VGLSA e trovate tutto. OVVIAMENTE vi prego di "vedere bene" come è fatto lo script

soffermandovi un attimino sulla create delle directory, dei Packages...insomma non lo eseguite ad occhi chiusi.

Prendiamo il flusso "**unita.csv**" presente della cartella per lui avremo:

	"TRENINO"		
Flusso	Tabelle "00"	Tabelle "01"	Tabelle "02"
unita.csv	GRP00_UNT	GRP01_UNT	GRP02_UNT
(FULL)	GRP00_UNTPER	GRP01_UNT LOG	
Note	<p>Sono due tab. esterne che puntano al flusso ed hanno tutti i campi "testo" per evitare che, in caso di dati sporchi sul flusso, vi siano delle "perdite" di righe (es: una data mal formattata). La "unt" serve alla procedura per prendere i record, la "per" (periodo) serve invece per recuperare il periodi di riferimento (es: 200909). Questa infatti verrà usata col rownum=1.</p>	<p>La "unt" è la tabella intermedia utilizzata per effettuare le validazioni sui dati. In particolare, per l'esempio delle unita, per validare che il codice unita sia numerico. La "log", invece, contiene gli scarti eventuali. La Log ovviamente è con le colonne "stringa" :) La "unt" viene gestita dal processo in modalità "truncate/insert" . Cioè ad ogni caricamento si svuota e si ricarica con i dati del mese in oggetto.</p>	<p>E la tabella finale del 1° livello sulla quale vengono messi i record validati presenti nella "01". Qui vi risiedono i dati pronti ad essere portati sul 2° livello nella "stellina". Per comodità abbiamo deciso di creare una partizione corrispondente ad un periodo caricato. In questo modo abbiamo (salvo scarti nei log) la "copia" del flusso del mese "x" sempre presente nella partizione "P_x". Ovviamente questo porta ad un grosso aumento dello spazio sul db in quanto i record vengono "replicati" su ogni partizione...quindi ogni tanto dovremo droppe delle partizioni per svecchiare il db. Questa gestione è comoda anche in caso di ri-carichi di una mensilità in quando andremo a fare una "drop partition" (ddl) e non una "delete" (dml) che potrebbe essere immensamente più lenta.</p>

Qualche informazione "generale" su questi campi "tecnici" :

ID_PER	Viene recuperato dalla prima "riga" dei flussi e rappresenta la "data di riferimento" alla quale si "riferiscono" i dati presenti nel flusso. Es: 200909.
--------	---

	Vuol quindi dire che "oggi stiamo caricando i dati di settembre 2009". Ricordo che il nostro motorino ha una frequenza MENSILE. Notate anche che il ID_PER viene utilizzato per la creazione delle PARTIZIONI delle tabelle "02" (es: P_200909).
ID_LOTTO	E' un semplice contatore dei "caricamenti" (lotti di caricamento) serve solo ai fini di controllo/audit e per interrogare più comodamente le tabelle di log. E' gestito da una sequence. Non so come dirlo, ma rappresenta il "numero della esecuzione".
D_INS	Lo troverete su tante tabelle e serve solo a registrare il "momento" nel quale il record viene inserito. E' un campo tecnico utile ai programmatori per gestire eventuali interventi manuali o controlli (es: quali sono i record caricati stanotte tra le due e le tre?) Nb: <i>D_UPD e D_DEL, nel caso, hanno lo stesso scopo per gestire momenti di aggiornamento e cancellazioni (logiche)<-- quelle fisiche il record non lo trovate più :)</i>

il processo di caricamento è governato dal "PACKAGE_ALIMENTAZIONE" il quale utilizza in modo DINAMICO le seguenti due tabelle di "lavoro/work" :

WRK ANAGRAFICA	Viene recuperato dalla prima "riga" dei flussi e rappresenta la "data di riferimento" alla quale si "riferiscono" i dati presenti nel flusso. Es: 200909. Vuol quindi dire che "oggi stiamo caricando i dati di settembre 2009". Ricordo che il nostro motorino ha una frequenza MENSILE. Notate anche che il ID_PER viene utilizzato per la creazione delle PARTIZIONI delle tabelle "02" (es: P_200909).
WRK CONFIGURAZIONE	<p>Premesso che:</p> <ul style="list-style-type: none"> - le logiche di processare i flussi nelle tabelle del trenino sono identiche. - volevamo evitare di fare una "procedura" per ogni flusso <p>Abbiamo deciso di fare una sola procedura GENERICA (che è il package_alimentazione_01) che si adattasse a tutti i flussi.</p> <p>Infatti il package fa un utilizzo pesante delle execute immediate (Dynamic SQL) andando a recuperare di volta in volta, le "parti" variabili che servono a comporre il comando da fare.</p> <p>Il recupero delle informazioni "variabili" viene fatto tramite semplici funzioncine tipo AD ESEMPIO "get_external_tab_name" che recupera il nome della tabella esterna necessaria per la "composizione" del comando sql.</p> <p>Questa parte del codice va studiata bene. Potete "vedere" il comando sql generato "run time" dal package andando</p>

	nella tabella di audit. In questo modo avete idea del "risultato" finale del comando che di volta in volta viene costruito.
--	---

controlli che vengono fatti sulla staging area:


PREMESSA	<p>Tipicamente sulla SA (o 1° livello) vengono fatti dei controlli per VALIDARE il dato ricevuto dai gestionali tramite i flussi di alimentazione.</p> <p>Ovviamente questi controlli VARIANO dal "cliente" e "dal progetto" ed è impossibile avere una "lista" completa. Volendo quindi stare GENERICI, riporto qui di seguito alcuni semplici controlli basilari:</p> <ul style="list-style-type: none"> . "non nullabilità" dei campi chiave (es: Cod. Unità) . correttezza dei tipi di dati (data type) che siano coerenti con quanto definito in fase di requisiti (coerenti col tracciato file). (es. un campo numerico sia un numero...un campo data sia formattato effettivamente come una data) . corretti valori di "range" nei campi (es. l'età sia maggiore di zero)
CONTROLLI	<p>Ad oggi sono implementati controlli relativi alla correttezza dei data type (un numero sia un numero e una data sia una data).</p> <p>Tali controlli vengono fatti tramite delle semplicissime funzioncine "f_is_a_valid_number" che, preso come input una stringa (es: '03') testano se questa sia effettivamente un numero (con la to_number) e restituisco 0 per errori e 1 se tutto ok.</p> <p>Le righe con "zeri" vanno nelle tabelle di scarti (le LOG) le righe con gli "uno" vanno nella tabella "01" per poi essere definitivamente portati nelle finali del 1° livello; le "02".</p>

VGLDM - 2° Livello - Data Mart

per creare lo schema basta collegarsi come system e fare copia/incolla ed eseguire lo script **02 CREADB VGLDM.sql**

DIMENSIONI

Origine 1° Liv.	Tabella	Gestione	Descrizione
<NULL>	D_PERIOD	<NULL>	<p>Statica:</p> <p>caricata "one shot" tramite un procedimento "manale". Contiene le informazioni relative al "calendario".</p> <p>Vedi esempio:</p>

			 caricamentoD_PERIO DO.pdf
GRP02_ANA	D_ANAGRAFICA_SCD	SCD	Si storicizza al variare della unita o della mansione (ruolo)
GRP02_RUO	D_RUOLI_MERGE	MERGE	Semplice merge
GRP02_UNT GRP02_ACR	D_UNITA_SCD	SCD	Si storicizza al variare del codice di accreditamento. Nb: su questa dimensione abbiamo effettuato la denormalizzazione dell'accreditamento all'interno dell'unità.

Note:

le tabelle dimensionali sopra elencate ci danno il senso di:

- dove (unità)
- chi (anagrafica)
- quando (periodo)
- "cosa" (in questo caso, potete "vederlo" come il ruolo)

Nei progetti reali, abbiamo mediamente almeno una decina di tabelle dimensionali...

Vi ricordo che nelle dimensioni abbiamo queste "particolarità":

SURROGATE KEY / ID

su tutte le tabelle dimensionali si va ad aggiungere una nuova chiave primaria numerica (in oracle gestita con le "sequence") che, "tutti", per abitudine chiamano "ID". Tale ID va ad "affiancare" la chiave che ci arriva dai flussi (la chiave del "gestionale")... che, per comodità, chiamiamo "CODICE". Quindi più o meno dovete avere in mente che nel data mart abbiamo "gli ID" e nei "flussi" (quindi ovviamente nella staging area) abbiamo i "CODICI".

Questo si fa per:

- **migliorare le performance** in quanto un indice su un "numero" è sicuramente più performante di un indice su un altro data type (es stringa o date). Siccome NON sappiamo cosa ci arriva dal "gestionale", noi del dwh, per sicurezza ci mettiamo un ID "a prescindere". Per "assurdo" una chiave ("codice") presente nel gestionale potrebbe anche essere una chiave composta da più campi (es: nome, cognome).

NB:

nelle tabelle gestite con la "merge" anche se noi usiamo gli id come chiave, i codici restano ugualmente "univoci". Mentre nelle tabelle "scd" sono solo gli id che identificano univocamente un record.

vedi questa immagine presa da wikipedia "scd (type 2)":

Supplier_Key	Supplier_Code	Supplier_Name	Supplier_State	Start_Date	End_Date
123	ABC	Acme Supply Co	CA	01-Jan-2000	21-Dec-2004
124	ABC	Acme Supply Co	IL	22-Dec-2004	NULL

(con la gestione scd ho più record "foto" per uno stesso soggetto e quindi "ABC" non è più univoco).

Fact Table

Origine 1° Liv.	Tabella	Gestione	Descrizione
PRESENZE	F_PRESENZE	INSERT	<p>La fact table da un punto di vista della "gestione" è sicuramente la più semplice in quando viene sempre alimentata in INSERT. Ovviamente, è delicata in quanto è quella che contiene sicuramente il maggior numero di record (anche milioni) ed è per questo che spesso viene partizionata per renderla più fruibile alle analisi. vi ricordo che la logica di partizionamento della tabella dei fatti deve essere scelta in base "al maggior numero di logiche di accesso ad essa".</p> <p>La fact table contiene quindi soltanto colonne numeriche... Gli ID che puntano alle dimensioni e i "numeri" relativi alle misure (i fatti) ad es. importi e quantità. Nel ns. esempio abbiamo le "ore lavorate".</p> <p>Come detto la Fact viene alimentata in insert.</p> <p>Qui sotto la tabella provo a simulare una pseudo query che carica una generica fact.</p>

Esempio.

ipotesi:

dimensioni: Tempo, Unita, Personale, Mansioni e per semplicità abbiamo solo la dimensione Unità gestita con la tecnica "scd". I fatti riportati sono le nostre ore del "flusso presenze" (per semplicità metto solo le "ore di presenza").

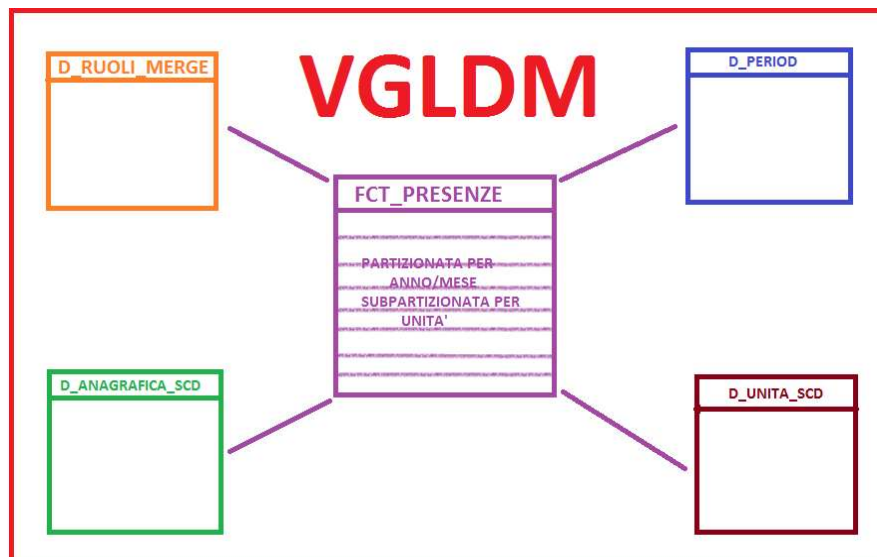
L'ipotetica select che la popola potrebbe essere qualcosa del genere:

```
"INSERT INTO FACT "
select b.IDTEMPO(*),c.IDUNITA, d.IDPERSONA, e.IDMANSIONE,
a.orelavorate
from "staging area".flussopresenze A, d_tempo B, d_unita_scd C,
d_personale D, d_mansioni E
where
a.giornopresenza = b.giorno AND
(a.codiceunita      = c.codiceunita and a.giornopresenza between
c.dal and nvl(c.al,'31/12/3999') ) AND
a.codicefiscale    = d.codicefiscale AND
a.codicemansione = e.codicemansione
```

l'IDTEMPO invece della sequence lo possiamo fare "parlante" es: 20160327
(annomesegiorno) tanto "numero è"...

VGLDM:STAR SCHEMA

Nel secondo livello lo scopo è quello di implementare lo StarSchema che ha come FactTable la tabella 'FCT_PRESENZE' partizionata per anno/mese e sub-partizionata per unità, e come dimensioni le tabelle '02' ottenute nel primo livello.



Come primo passo si creano le viste che popoleranno le dimensioni a partire dal livello VGLSA.

Si osservi che le dimensioni 'D_UNIA_SCD' e 'D_ANAGRAFICA_SCD' sono popolate utilizzando la *SlowlyChangingDimension(scd)*, mentre la dimensione 'D_RUOLI_MERGE' utilizzando la *merge*.

In particolare per il caricamento delle dimensioni 'D_ANAGRAFICA_SCD' e 'D_UNITA_SCD' il Package Alimentazione richiama due procedure appositamente create per il popolamento mediante la *SlowlyChangingDimension(scd)*.

La dimensione D_PERIOD viene popolata con un'apposita procedura e tiene traccia del periodo dal 2009 al 2015.

VGLDM: TABELLE WORK

Nel processo di caricamento della FactTable, ci serviamo nuovamente di apposite tabelle di work e configurazione:

- **WRK_ANAGRAFICA_FLUSSI**: E' riportato il val_tab_name che mi determina il nome della dimensione, il flag che indica se il flusso è disponibile, il flag per l'ordine di caricamento dei flussi, una nota_lista_flussi che mi indica da quali flussi è composta la dimensione e una nota_lista_view che mi indica da quale vista viene popolata la dimensione.
- **WRK_AUDIT_CARICAMENTI**: Tiene traccia dei caricamenti.
- **WRK_AUDIT_LOG**: Tiene traccia di tutti i passi durante il caricamento, riportando messaggi di errore o di avvenuta esecuzione.
- **WRK_CONFIGURAZIONE**: Sono riportate in essa tutte le configurazioni necessarie al caricamento delle facttable.

VGLDM: PACKAGE

Per il completamento del secondo livello (VGLdm) sono stati implementati due diversi Package:

1. **Package PKG_UTILS_01**: Nel quale sono implementate tutte le procedure e le funzioni utili al package Alimentazione per il caricamento della facttable e delle dimensioni.
2. **Package PKG_ALIMENTAZIONE_01**: Grazie ad un cursore sulla tabella WRK_ANAGRAFICA_FLUSSI carica tutti i flussi che hanno il flag_disponibile uguale a 'Y', in tre step
 - Nel primo step controlla se esiste la partizione relativa a quel periodo, se esiste la cancella e la ricrea altrimenti la crea;
 - Nel secondostep popola la tabella delle dimensioni e la facttable;
 - Nel terzo step analizza la tabella.

SCRIPT

Di default il flag 'FLG_DISPONIBILE' della work_anagrafica_flussi è settato su 'N'.

Questo è stato fatto per rendere più dinamico il caricamento dei flussi mediante l'utilizzo di uno script unix.

Tale script, nella prima parte, legge i flussi pervenuti nelle apposite cartelle (ad esempio il flusso accreditalenti.csv, se è stato ricevuto, si troverà nella cartella COGE), e inserisce il filename di tali flussi in un file chiamato 'elenco_flussi'.

Nella seconda parte lancia una funzione PL/SQL che setta il flag 'Flg_disponibile' della tabella wrk_anagrafica_flussi su 'Y' se il codice flusso è presente nel file 'elenco_flussi'.

Si osservi che il file "check_files_disponibili.sh" deve essere in formato unix per essere lanciato. Quindi se dovessero esserci problemi aprire il file su NOTEPAD++, andare su Modifica e selezionare 'Formato UNIX' in 'Converti Carattere di Fine Linea'.