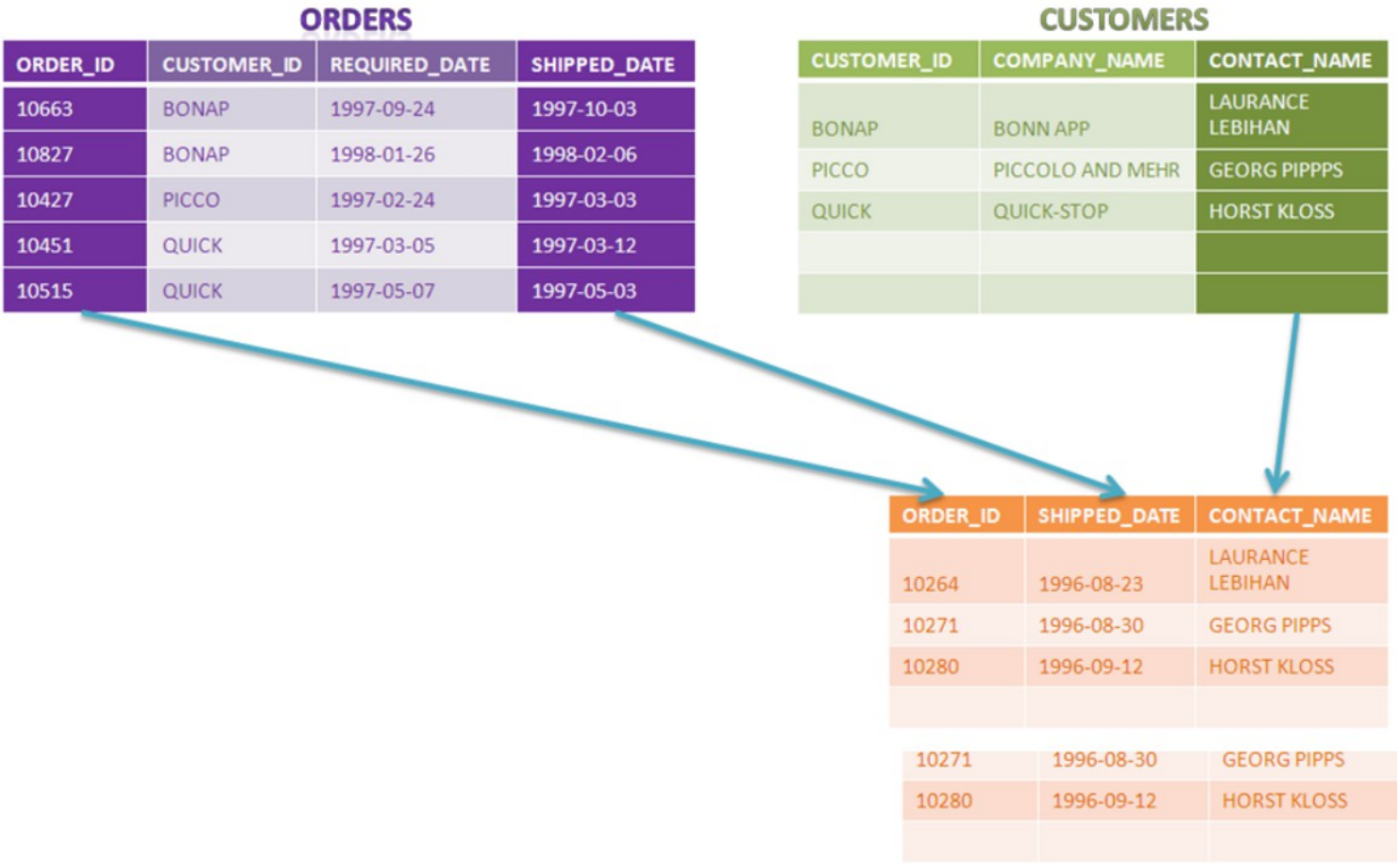
Viste



Cos'e' ?

La vista *(view)* è una tabella virtuale: non esiste realmente anche se l'utente la percepisce sotto forma di tabella.

Una tabella di base (al contrario di una view) ha un riscontro fisico: ad ogni elemento della tabella corrisponde un elemento sul supporto magnetico.

La definizione di una view è basata su tabelle di base

UTILITÀ DELLE VISTE

- Per nascondere certe modifiche all'organizzazione logica dei dati (indipendenza logica)
- Per offrire visioni diverse degli stessi dati senza ricorrere a duplicazioni
- Per far funzionare applicativi dopo aggiustamenti/modifiche delle tabelle usate (es. tabella articoli in il campo 'categoria' diventa una nuova tabella)

- **Per rendere più semplici, o per rendere possibili**, alcune interrogazioni

Le viste, fra le altre cose, **rendono piu semplice la stesura delle query proprio come le funzioni rendono piu semplice la stesura dei programmi.**

Quando abbiamo un compito complesso da risolvere lo dividiamo in sottocompiti (dividi et impera)
Operiamo così nella programmazione strutturata

In un modello ER, quando abbiamo un certo numero di tabelle, ci si trova ad affrontare query la cui risoluzione non è sempre immediata.
A volte si richiede di effettuare join su 4,5 o piu tabelle, il tutto magari con select annidate.
In questo caso conviene crearci opportune viste che ci permettono di arrivare gradualmente alla risoluzione della query complessa.

Molte volte la vista che unisce i**n un' unica join tutte le tabelle** è la **vista base** su cui lavorare.

SINTASSI DELLA VISTA

Nella maggioranza dei casi si creano viste che permettono una visione unitaria della base di dati,
cioè come SE TUTTE LE TABELLE FOSSERO RAGGRUPPATE in un'unica tabella.

Esempio su una sola tabella → Definire una vista IMPIEGATI5 che contiene tutti e soli gli impiegati del dipartimento 5 che guadagnano piu' di 10000 euro.
(la tabella IMPIEGATO ha attributi: **CF**, COGNOME, STIPENDIO).

```
CREATE VIEW IMPIEGATI5(CF5, NOME5, COGNOME5, STIP5) AS
SELECT CF, NOME, COGNOME, STIPENDIO
FROM IMPIEGATO
WHERE DIP = 5 AND STIPENDIO >10000
```

Vista della vista... e' possibile? → Certo!

Esempio: A partire dalla vista IMPIEGATI5, puo' essere costruita una seconda vista che contiene tutti e soli gli impiegati del dipartimento 5 con uno stipendio compreso tra 10000 e 25000 (impiegati poveri???? ;)

```
CREATE VIEW IMPIEGATI5POVERI AS
SELECT -- NB: non ho definito nuovi nomi di campo
FROM IMPIEGATI5
WHERE STIPENDIO <25000
```

ESEMPIO GENERALE: (vista con N campi su M tabelle unite da join):

```
CREATE VIEW nome_della_vista
(nome_nuovo_campo1,nome_nuovo_campo2,...nome_nuovo_campoN) AS
SELECT campo1, campo2,..., campoN
FROM tabella1,tabella2...,tabellaM -- NB: tabelle o anche viste
WHERE join_tra_M_tabelle
```

Questa qui sopra è la sintassi della vista generale su piu' tabelle/viste. Cosa ci permette di fare la vista?

La vista permette di ottenere **una nuova tabella virtuale**, in sola visualizzazione (operiamo sulla vista con la sola SELECT)

Perchè la vista è una tabella virtuale?

Perchè nella base di dati **non viene creata nessuna nuova tabella con CREATE VIEW**, ma a tutti gli effetti per quanto riguarda le query SQL possiamo operare sulla lista come se fosse una tabella reale.

COME PREPARARSI LA VISTA … per gradi:

conviene:
1) preparare prima la select contenuta nella vista (con una select normale)
2) poi trasformare la select in una vista

Approfondimenti sulle viste:

Se i nomi delle colonne della vista non sono specificati, **sono utilizzati quelli specificati nella select.**

I nomi delle colonne devono essere obbligatoriamente specificati se:

– rappresentano il risultato di una funzione interna
– rappresentano il risultato di un' espressione o sono costanti
– due colonne hanno lo stesso nome

Cancellazione di viste

Per cancellare una vista:

DROP VIEW nome-vista;

La cancellazione di una tabella (di base) comporta l'eliminazione automatica delle viste che ad essa si riferiscono

Aggiornamento dei dati nelle viste (in generale NON possibile)

• E` possibile eseguire operazioni di aggiornamento dei dati contenuti in una vista solo per alcune tipologie di viste.

• Standard SQL-92: sono aggiornabili le viste in cui una sola riga di ciascuna tabella di base corrisponde a una sola riga della vista.

Queste viste sono dette *intrinsecamente aggiornabili.*

Vedi sotto un esempio/esercizio sulle viste: CALCIATORI/SQUADRE

```
CREATE TABLE squadra (
codice          CHAR    ( 6) primary key,
nome                    VARCHAR (30)
);

CREATE TABLE calciatore  (
codice          INTEGER    primary key,
squadra                 CHAR    (6) REFERENCES squadra(codice),
cognome                 VARCHAR (20),
nome                    VARCHAR (20),
data_nascita    DATE,
stipendio               DECIMAL (10)
);

INSERT INTO squadra VALUES ('S00001', 'juventus FC');
INSERT INTO squadra VALUES ('S00002', 'internazionale');
INSERT INTO squadra VALUES ('S00003', 'InterMilan');

INSERT INTO calciatore VALUES (1, 'S00001', 'Cortesi', 'Geppeo', '11-3-1996', 30);
INSERT INTO calciatore VALUES (2, 'S00001', 'Gigi', 'Marca', '12-2-1995', 27);
INSERT INTO calciatore VALUES (3, 'S00001', 'Blu', 'Salvatore', '12-6-1997', 26);
INSERT INTO calciatore VALUES (4, 'S00002', 'Pierotti', 'Michelangelo', '11-6-1994', 57);
INSERT INTO calciatore VALUES (5, 'S00002', 'Mastrota', 'Giorgio', '2-10-1983', 17);
INSERT INTO calciatore VALUES (6, 'S00002', 'Viola', 'Paolo', '1-2-1980', 45);
INSERT INTO calciatore VALUES (7, 'S00003', 'Rossi', 'Tommaso', '5-9-1993', 21);
INSERT INTO calciatore VALUES (8, 'S00003', 'Verdi', 'Bruno', '3-8-1992', 89);
INSERT INTO calciatore VALUES (9, 'S00003', 'Capaldi', 'Pietro', '4-9-1985', 56);

//A)Tutti i calciatori, con la loro squadra
SELECT cognome, calciatore.nome, squadra.nome
FROM calciatore, squadra
WHERE calciatore.squadra = squadra.codice
ORDER BY squadra.nome, calciatore.cognome, calciatore.nome

//B) Tutti i calciatori, con la loro squadra di appartenenza, che hanno uno stipendio
mensile superiore a una certa cifra
SELECT calciatore.nome, calciatore.cognome, squadra.nome, calciatore.stipendio/12 AS
stipendioMensile
FROM squadra, calciatore
WHERE stipendio/12 > 3
AND squadra.codice = calciatore.squadra
ORDER BY cognome, calciatore.nome;
```

// Con VISTA:

```
CREATE VIEW CS ( CS_cognome, CS_nome, CS_squadra, CS_stipendio)AS
SELECT cognome, calciatore.nome,  squadra.nome, stipendio
FROM calciatore, squadra
WHERE calciatore.squadra = squadra.codice

// Select su view
SELECT CS_nome, CS_cognome, CS_squadra, CS_stipendio/12
FROM CS
WHERE CS_stipendio/12 > 3
ORDER BY CS_cognome, CS_nome
```

//Esercizi:
//Riscrivi con viste i seguenti:

```
// C) Le squadre che,  in un anno spendono per i loro calciatori (per gli stipendi) piu' di
una certa cifra

SELECT SUM(stipendio) AS cifraSpesa, squadra.nome
FROM squadra, calciatore
WHERE squadra.codice = calciatore.squadra
GROUP BY squadra.nome
HAVING SUM(stipendio) > 110
ORDER BY cifraSpesa;

//D) Il calciatore piu’ giovane
SELECT squadra.nome,cognome, calciatore.nome, data_nascita, AGE(data_nascita) AS eta
FROM calciatore,squadra
WHERE calciatore.squadra = squadra.codice AND AGE(data_nascita) = (SELECT
min(AGE(data_nascita)) FROM calciatore)

//E) La situazione spesa annuale delle squadre per i loro calciatori, riportante:
Nome Squadra, Spesa annuale, Differenza fra la sua spesa e la spesa media delle squadre
```