**CAPITOLO 4**

**Esercizi riepilogativi**

Esercizio 1

Dato il seguente schema relazione che descrive il calendario di una manifestazione sportiva a squadre nazionali:

STADIO(Nome, Città, Capienza)

INCONTRO(NomeStadio, Data, Ora, Squadra1, Squadra2)

NAZIONALE(Paese, Continente, Categoria)

Esprimere in SQL le seguenti interrogazioni:

1. Estrarre i nomi degli stadi in cui non gioca nessuna nazionale europea

Immagine che contiene testo, schermata, Carattere

Descrizione generata automaticamente

1. Esprimere l’interrogazione in algebra relazionale, in calcolo e in Datalog.
2. Estrarre la capienza complessiva degli stadi in cui si giocano le partite che hanno come prima squadra una squadra sudamericana( per semplicità si conteranno più volte le capienze degli stadi in cui si giocano più partite)

Immagine che contiene testo, schermata, Carattere, linea

Descrizione generata automaticamente

L’errore che qui avevo fatto è stato quello di seguire il procedimento utilizzato nel passo (a). Secondo me poteva comunque funzionare qui, ma applicare una funzione di join sicuramente è più efficiente. Infatti da quello che ricordo il DBMS preferisce sempre eseguire operazioni di join rispetto alle altre.

Esercizio 2

Dato il seguente schema relazionale:

Moto(Targa, Cilindrata, Marca, Nazione, Tasse)

Proprietario(Nome, Targa)

1. Estrarre i nomi dei proprietari di solo moto giapponesi di almeno due marche diverse

Immagine che contiene testo, schermata, Carattere, linea

Descrizione generata automaticamente

Anche qui ho sbagliato un po’ di cose e per questo motivo cercherò di soffermarmi su queste facendo chiarezza per le volte future:

Per prima cosa non ho pensato al join, successivamente ho sbagliato a mettere l’argomento della clausola Where.

Sostanzialmente vado a selezionare i nomi di quei proprietari che non compaiono nella tabella che contiene i nomi dei proprietari che presentano un nazionalità della moto diversa da Giappone. In questo modo utilizzando “NOT IN” vado a ricavare una tabella che contenga i nomi dei proprietari che possiedono solo moto di nazionalità Giapponese.

A questo punto ricordiamo che il join fa in modo che il nome del proprietario si ripeta più volte e per questo motivo utilizziamo la clausola GROUP BY che consente di creare un sottoinsieme di una tabella originaria andando a selezionare le righe che presentano lo stesso valore su un attributo. Ricorda che quando è presente il GROUP BY, prima viene eseguita l’interrogazione come se non ci fosse e successivamente viene eseguito il raggruppamento.

Qui non avrei potuto utilizzare il “DISTINCT Nome” in quanto non avrebbe avuto senso! Questo costrutto va ad eliminare le righe per cui si hanno doppioni sullo stesso valore dell’attributo Nome eliminando così tutte le informazioni che certificano l’appartenenza di un proprietario di più moto.

**Esercizi**

Esercizio 4.1

Ordinare i seguenti domini in base al valore massimo rappresentabile, supponendo che integer abbia una rappresentazione a 32 bit e smallint a 16 bit: numeric (12,4), decimal (10), decimal (9), integer, smallint, decimal (6,1)

SmallInt

Integer

Decimal (10)

Decimal (9) 🡪 questa rappresentazione è errata

Numeric (12,4)

Decimal(6,1)

SmallInt: si usano per rappresentazioni binarie standard e possono rappresentare numeri che vanno da . In questo caso, con l’utilizzo di 16 bit rappresentativi, potremmo quindi andare a rappresentare numeri che vanno da (approssimativamente -32768 a 32767)

Integer: stessa cosa per integer che anche lui viene utilizzato per rappresentazioni binarie, utilizzando in questo caso 32 bit rappresentativi, potremmo andare a rappresentare numeri che vanno da (approssimativamente da -2147483648 a + 2147483647)

Decimal(10): questo tipo numerale viene utilizzato come gli altri per la rapresentazione di numeri esatti e prevede la specifica di una precisione e opzionalmente una scala che quantifica il numero di cifre che dovranno essere presenti dopo la virgola. In questo caso il nostro tipo numerale consentirà la rappresentazione di 10 cifre, quindi da -9.999.999.999 a + 9.999.999.999 .

Decimal(6,1): tengo a specificare anche questo decimale, anche se sopra lo abbiamo ampiamente descritto, in quanto oltre alla specifica della precisione che indica il numero di cifre significative per descrivere il numero, la scala contribuisce a specificare il numero di cifre che dovranno comporre la parte frazionaria. Esso può quindi rappresentare un numero con un massimo di cinque cifre prima del punto decimale. (circa 10^5 – 0.1)

Numeric(12,4): credo che valga lo stesso discorso fatto per il decimal ma qui abbiamo valori differenti sulla precisione e sulla scala, quindi potremmo al massimo rappresentare un numero con 8 cifre prima del punto decimale

Quindi l’ordinamento corretto è il seguente:

Decimal(10)

Integer(32bit)

Decimal(9)

Numeric(12,4)

Decimal(6,1)

SmallInt(16bit)

Esercizio 4.2

Definire un attributo che permetta di rappresentare stringhe di lunghezza massima pari a 256 caratteri, sui cui non sono ammessi valori nulli e con valore di default “sconosciuto”

Immagine che contiene testo, schermata, Carattere, linea

Descrizione generata automaticamente

Esercizio 4.3

Dare le definizioni in SQL delle tre tabelle Fondista(Nome, Nazione, Età), Gareggia(NomeFondista, NomeGara, Piazzamento) e Gara(Nome, Luogo, Nazione, Lunghezza), rappresentando in particolare i vincoli di foreign key della tabella Gareggia.

Immagine che contiene testo, schermata, Carattere, linea

Descrizione generata automaticamente

Immagine che contiene testo, schermata, Carattere, linea

Descrizione generata automaticamente

Immagine che contiene testo, schermata, Carattere, linea

Descrizione generata automaticamente

La cosa che qui è molto importante sono i vincoli interelazionali presenti che sono i vincoli di integrità referenziale che compaiono fra gli attributi NomeFondista e NomeGara della tabella Gareggia e gli attributi Nome di Fondista e Nome di Gara.

L’unico vincolo che impone il vincolo di integrità referenziale è quello di far in modo che gli attributi referenziati nella tabella esterna siano unici, ovvero contrassegnati con il vincolo UNIQUE.

Questo perchà ad ogni valore dell’attributo presente nella tabella interna dfevo corrisapondere per forza uno e un solo valore nel corrispettivo attributo nella tabella esterna.

Successivamente si possono avere due opzioni per rappresentare il collegamento di un attributo con uno di un’altra tabella.

Immagine che contiene testo, schermata, Carattere

Descrizione generata automaticamente

Solitamente il costrutto FOREIGN KEY (attributoEst1, attributoEst2) REFERENCES nomeTabellaInt(AttributoInt1,AttributoInt2) viene utilizzato quando si hanno un insieme di attributi legati da un vincolo.

Quando invece bisogna descrivere un singolo legame basta riportare dopo la definizione dell’attributo REFERENCES tabellaInt(attributoInt).

In questo caso non so perché gli attributi referenziati provengono da due tabelle differenti e non so quale rappresentazione sia più conveniente tra le due proposte, se c’è quindi una preferenza o un modo più compatto per descrivere lo stesso legame.

Esercizio 4.4

Dare le definizioni di SQL delle tabelle AUTORE(Nome,Cognome, DataNascita, Nazionalità),

LIBRO(TitoloLibro, NomeAutore, CognomeAutore, Lingua). Per il vincolo di foreign key specificare una politica di cascade sulle cancellazioni e di set null sulle modifiche.

Immagine che contiene testo, schermata, Carattere

Descrizione generata automaticamente

Questa è la prima tabella, ovviamente non occorre aggiungere vincoli di non nullità o di unicità perché sono gia compresi indirettamente nella definizione di chiave principale. Essa presenta alcune caratteristiche infatti come il fatto che gli attributi che costituiscono la chiave non possono assumere valori nulli altrimenti non si renderebbe impossibile identificare la tupla e il vincolo di unicità è sottointeso per il fatto che la chiave è nota per identificare univocamente una tupla pertanto non possono essere presenti due tuple con gli stessi valori sugli attributi identificanti.

Immagine che contiene testo, schermata, Carattere, linea

Descrizione generata automaticamente

Con il comando FOREIGN KEY sto creando un vincolo di integrità referenziale fra gli attributi NomeAutore e CognomeAutore di Libro rispettivamente con gli attributi di Nome, Cognome di Autore. RICORDA che questa cosa è valida solo se Nome e Cognome della tabella interna Autore formano una chiave primaria o una chiave unica.

Con il comando on DELETE CASCADE stiamo indicando la scelta risolutiva in caso di violazioni di integrità dovute ad eliminazioni di valori nella tabella esterna. In particolare stiamo dicendo al nostro DBMS di gestire queste eliminazioni di valori per gli attributi NomeAutore e CognomeAutore andando ad eliminare tutte le righe della tabella interna e delle tabelle associate in cui si presentavano questi valori eliminati.

Con il comando on UPDATE SET NULL, stiamo dicendo al DBMS di impostare al valore NULL i valori nella tabella interna degli attributi corrispondenti modificati nelle tabelle esterne.

Esercizio 4.5

Dato lo schema dell’esercizio precedente, spiegare cosa può capitare con l’esecuzione dei seguenti comandi di aggiornamento:

1. *DELETE FROM Autore*

*WHERE Cognome = “Rossi”*

Questo comando è un comando di cancellazione che consente di cancellare dalla tabella Autore tutte quelle righe che presentano come valore dell’attributo Cognome, Rossi.

Nel caso in cui non fosse stato indicata la condizione dopo la clausola WHERE si sarebbe considerato WHERE TRUE e sarebbero state eliminate tutte le righe della tabella.

1. **UPDATE Libro set Nome = “Rossi”**

Questo comando permette di eseguire un’operazione di modifica sull’attributo Nome dell’entità Libro impostandola a Rossi. Non essendo presente la clausola where suppongo che la modifica operi su tutte le righe della tabella e non su una in particolare.

1. **UPDATE Libro set Nome = “Umberto”**

**WHERE Cognome = “Eco”**

Questo comando invece, differentemente da quello precedente, presenta un clausola WHERE. La modifica viene effettuata sull’attributo Nome dell’attributo Libro e in particolare verrà cambiato il nome in “Umberto” dove il cognome dell’autore è “Rossi”.

1. **INSERT into Autore(Nome,Cognome)**

**values (“Antonio”, “Bianchi”)**

Il seguente comando rappresenta un inserimento di due valori per due attributi Nome e Cognome nella tabella Autore. Viene in questo caso utilizzata la prima sintassi che prevede la specifica, tra parentesi tonde, dei valori da inserire.Questa forma permette di inserire singole righe. Gli attributi coinvolti allo stesso modo sono inseriti tra parentesi. L’ordine degli attributi con i relativi valori è essenziale, il primo attributo assumerà il primo valore specificato e così via!

Esistono comunque due forme in totale per esprimere un inserimento, ognuna utilizzata per uno specifico scopo. Solitamente la prima viene utilizzata per le azioni di inserimento di valori da parte degli utenti.

La seconda forma invece viene utilizzata per inserire un insieme di righe all’interno di una tabella e spesso utile per inserire informazioni in una tabella già presenti nel database.

1. **UPDATE Autore set Nome = “Italo”**

**WHERE Cognome = “Calvino”**

Similmente ad un caso già visto, questo comando consente la modifica del valore dell’attributo Nome dell’entità Autore nella riga in cui il cognome dell’autore è Calvino.

Esercizio 4.6

***Create domain Dominio integer default 10***

Rappresenta la sintassi per la creazione di un dominio personalizzato che è rappresentato da un tipo numerico Integer e viene successivamente specificato un valore di default fissato a 10, che viene utilizzato nel caso in cui non venga specificato nessun valore per tale campo. QUINDI SE IO NON INSERISCO NESSUN VALORE PER QUEL CAMPO INSERISCE IL VALORE DI DEFAULT?? E ALLORE A COSA SERVE IL VALORE DI DEFAUL DEFINITO PER L’ATTRIBUTO??

***Create table Tabella(Attributo Dominio default 5)***

Questo costrutto mi permette di creare una tabella all’interno del mio schema del database e di inserire all’interno un attributo con nome Attributo e che ha come dominio il dominio da noi prima costruito(esso rappresenterà comunque numeri interi) e nel caso in cui non fornissimo nessun valore per l’attributo esso assumerà il valore di default che sarà pari a 5. (QUANDO ENTRA IN GIOCO IL VALORE DI DEFAULT PER IL DOMINIO).

Indicare adesso cosa avviene in seguito a questi comandi:

***alter table Tabella***

***alter column Attributo drop defaul***

Questo comando permette di modificare la tabella Tabella e in particolare la colonna Attributo, eliminando il suo attributo di default.

***Alter domain Dominio drop default***

Permette di modificare il dominio Dominio andando ad eliminare il valore di default impostato pari a 10.

***Drop domain Dominio***

Mentre il comando “alter” effettua delle modifiche sui domini e sulle tabelle, il comando drop permette di rimuovere dei componenti che possono essere schemi, domini, tabelle, viste o asserzioni.

La sua sintassi è la seguente:

Drop <schema | table | column | view | assertion > NomeElemento [restrict | cascade ]

Dopo aver fatto questo breve riassunto, possiamo dire che il comando sopra va ad eliminare il dominio che presenta il nome “Dominio” solo se non appare in nessuna definizione nello schema.

Ovvero si sta applicando di default, visto che non è stata specificata, l’opzione *restrict.*

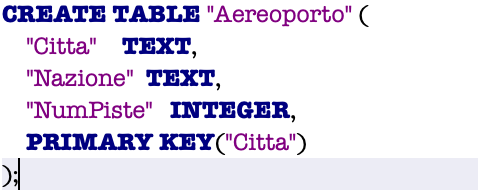
Esercizio 4.7

Dato il seguente schema:

AEREOPORTO (Città, Nazione, NumPiste),

VOLO (IdVolo, GiornoSettG, CittàPart, OraPart, CittàArr,OraArr, TipoAereo),

AEREO (TipoAereo, NumPasseggeri, QtaMerci)

Immagine che contiene testo, Carattere, schermata

Descrizione generata automaticamente

Immagine che contiene testo, schermata, Carattere

Descrizione generata automaticamente

1. Le città con un aereoporto i cui non è noto il numero di piste

Immagine che contiene testo, schermata, Carattere

Descrizione generata automaticamente

1. Le nazioni da cui parte e arriva il volo con codice AZ274

SELECT Nazione

FROM Aereoporto

WHERE Citta in (SELECT CittaArr

FROM Volo

WHERE IdVolo = "AZ274")

or (SELECT CittaPart

FROM Volo

Where IdVolo = "AZ274")

Questa è la query che ho scritto, ma ci sono alcuni problemi che devono essere risolti:

* L’utilizzo dei doppi apici potrebbe portare confusione e sarebbe corretto utilizzare gli apici singoli, in quanto convenzione per il linguaggio SQL
* La struttura della subquery non è corretta: non è inclusa in nessuna clausola “IN” o “EXIST” e questo causerà un errore perché SQL si aspetterebbe una condizione booleana dopo l’OR, ma invece trova una subquery non quantificata.
* Confronto diretta fra le subquery: si sta cercando di andare a confrontare direttamente la Citta con il risultato di due subquery, tuttavia per farlo entrambe le subquery dovrebbero restituire un singolo valore e il confronto deve essere fatto usando ‘IN’ o un operatore simile per entrambe le condizioni e non solo per la prima.

Ecco la query corretta:

Immagine che contiene testo, schermata, Carattere

Descrizione generata automaticamente

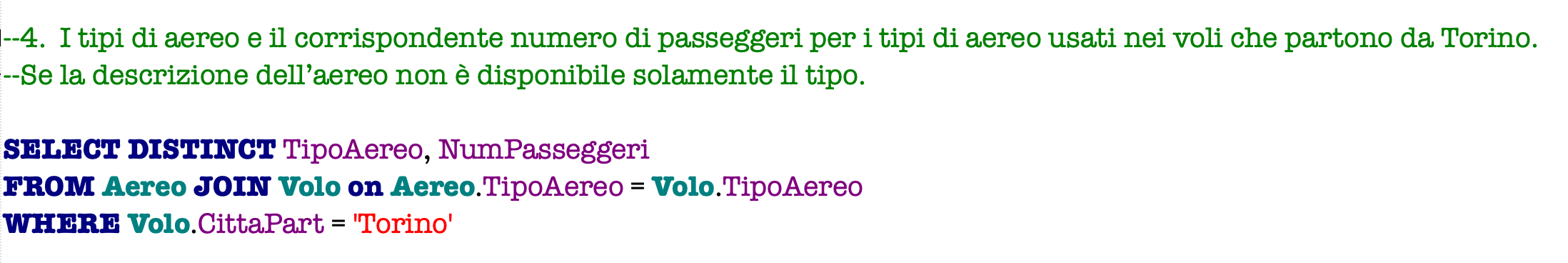
1. i tipi di aereo usati nei voli che partono da Torino

Immagine che contiene testo, Carattere, schermata

Descrizione generata automaticamente

Importante qui utilizzare DISTINCT che permette di eliminare eventuali doppioni di valori sull’attributo TipoAereo.

1. I tipi di aereo e il corrispondente numero di passeggeri per i tipi di aereo usati nei voli che partono da Torino. Se la descrizione dell’aereo non è disponibile solamente il tipo.



Qui ho un po di dubbi, specificatamente sul risultato nel caso in cui il numPasseggeri non sia presente come la quanrtità della merce. Basta cosi????? FORSE QUESTA COSA è DOBUTA AL FATTO CHE STO APPLICANDO UN JOIN CHE ESCLUDFE VALORI NULLI????

1. Le città da cui partono voli internazionali

Immagine che contiene testo, schermata, Carattere, linea

Descrizione generata automaticamente

Diciamo che questa quey per oraè stata la più difficile, ma finalmente dovremmo aver trovato la soluzione. La soluzione prevede di mostrare i nomi di città di partenza, senza ripetizioni, per dei voli che siano internazionali, ovvero parto da città di una nazione e atterrano in un città di una nazione defferente. Per mostrare questo cocnetto ho utilizzato due join differenti per poter confrontare le nazioni delle città di partenza con quelle di arrivo. Ho utilizzato una subquery per andare a confrontarle e un operatore di confronto < > con l’utilizzo del selettore any.

Questo fa in modo che la condizione possa essere verificata trovando una effettiva corrispondenza veritiera con al minimo una riga della tabella prodotta dalla subquery. Questo procedimento confrota il valore dell’attributo Nazione della città di partenza e Il valore della nazione della Città di arrivo(con la condizione che si sia parlando della stessa tratta, ovvero che si parte da A e si giunga a B).

1. Le città da cui partono voli diretti a bologna, ordinate alfabeticamente;

Immagine che contiene testo, Carattere, schermata

Descrizione generata automaticamente

Ho effettuato un join e imposto una clausola che selezionasse solo quelle righe della tabella che presentassero come Citta di arrivo Bologna. Successivamente ho ordinato le città in base al loro nome crescente.

1. Il numero di voli internazionali che partono il giovedi da Napoli

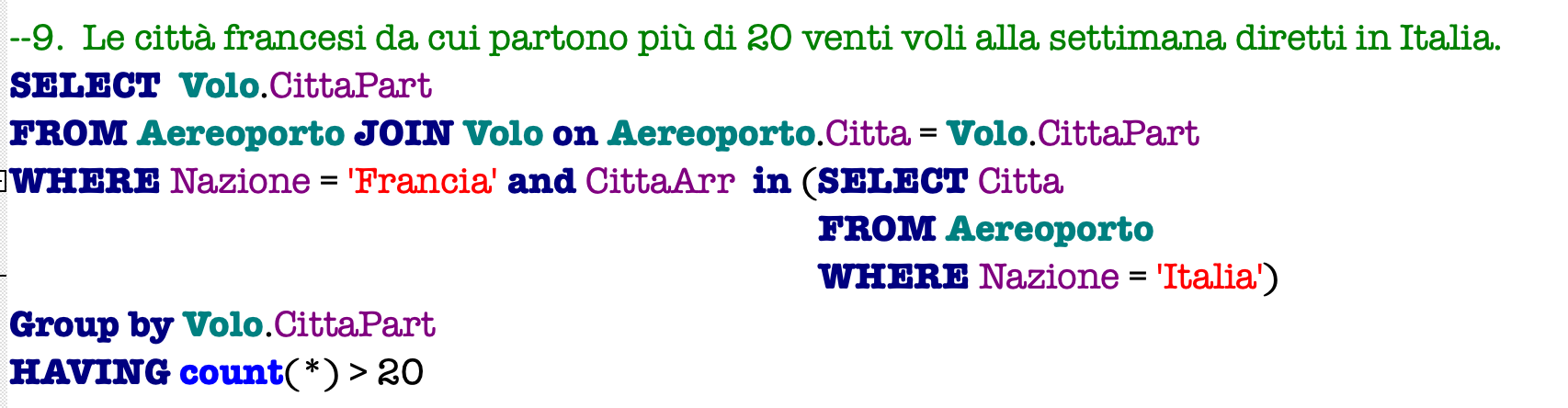
Immagine che contiene testo, schermata, Carattere

Descrizione generata automaticamente

1. Il numero di voli internazionali che partono ogni settimana da città Italiane(farlo in due modi, facendo comparire o meno nel risultato gli aereoporti senza voli internazionali)



1. Le città francesi da cui partono più di 20 venti voli alla settimana diretti in Italia.



1. Gli aereoporti italiani che hanno solo voli interni. Rappresentare questa interrogazione in quattro modi:
2. Con operatori insiemistici
3. Con una interrogazione nidificata con l’operatore not in
4. Con una interrogazione nidificata con l’operatore not exist
5. Con l’outer join e l’operatore di conteggio

Esprimere l’espressione pure in algebra relazionale

ii) SBAGLIATA MA PERCHE?Immagine che contiene testo, schermata, Carattere, linea

Descrizione generata automaticamenteD

Immagine che contiene testo, schermata, Carattere, linea

Descrizione generata automaticamente

GIUSTA?

(iii)

1. Le città che sono servite dall’aereo caratterrizzato dal massimo numero di passeggeri.

Immagine che contiene testo, schermata, Carattere, linea

Descrizione generata automaticamente

Esercizio 4.8

Dato il seguente schema:

DISCO(NroSerie, TitoloAlbum, Anno, Prezzo)

CONTIENE (NroSerieDisco, CodiceReg, NroProgr)

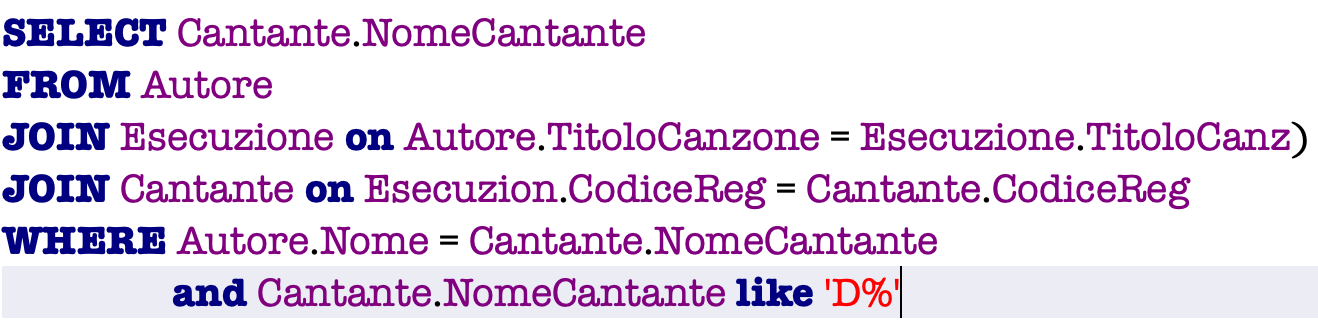
ESECUZIONE(CodiceReg, TitoloCanz, Anno)

AUTORE(Nome, TitoloCanzone)

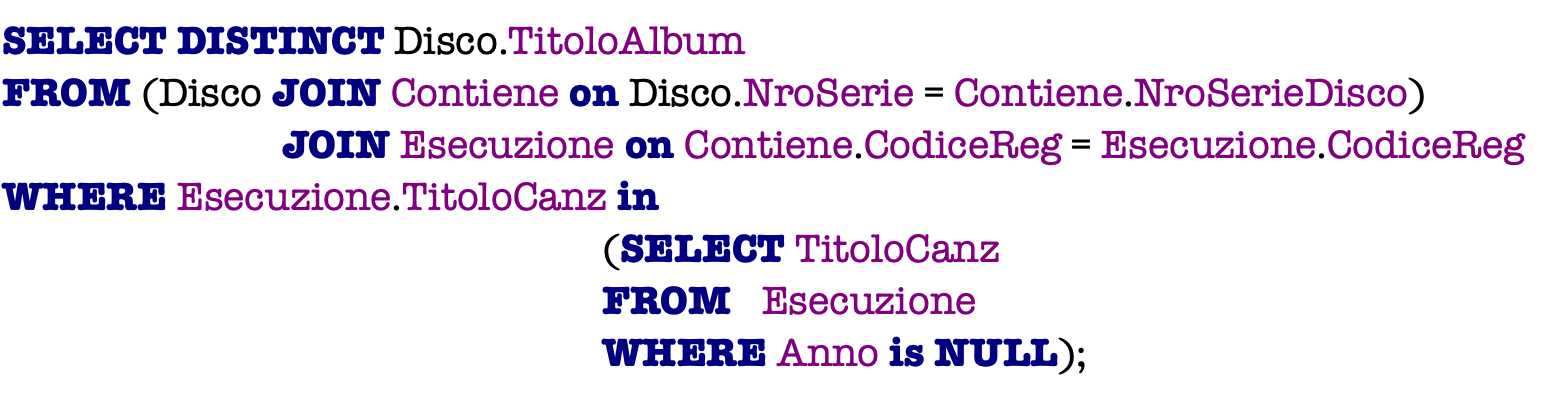
CANTANTE(NomeCantante, CodiceReg)

Formulare le interrogazioni che permettono di determinare:

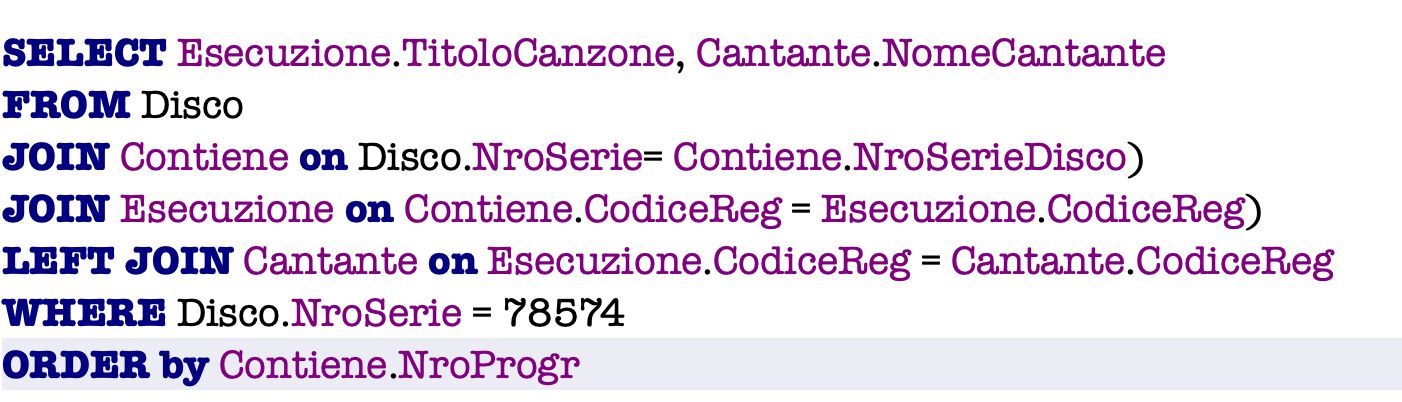
1. I cantautori (persone che hanno scritto e cantato la stessa canzone) il cui nome inizia per ‘D’



1. I titoli dei dischi che contengono canzoni di cui non si conosce l’anno di registrazione



1. I pezzi del disco con numero di serie 78574, ordinati per numero progressivo, con indicazione degli interpreti per i pezzi che hanno associato un cantante.



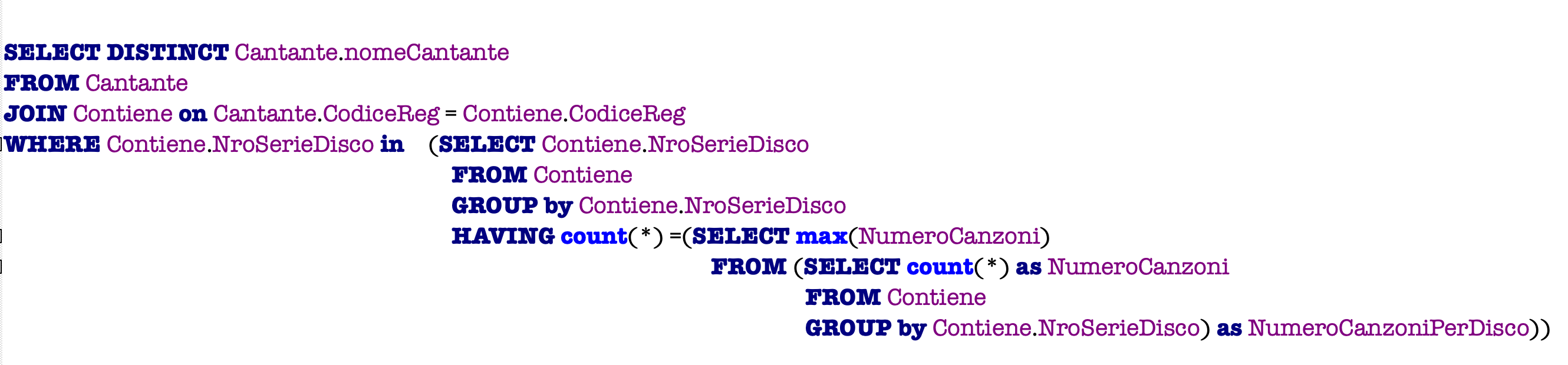
E’ importante qui notare che è necessario applicare un left join che permette di tenere in considerazioni anche quelle canzoni che presentano un cantante ‘sconosciuto’. La tabella risultano in questo caso, nell’attributo NomeCantante presenterà anche valori nulli.

1. Gli autori e cantanti puri, ovvero autori che non hanno mai registrato una canzone e cantanti che non hanno mai scritto una canzone

Immagine che contiene testo, schermata, Carattere, linea

Descrizione generata automaticamente

1. I Cantanti del disco che contiene il numero maggiore di canzoni



Query 1:

Immagine che contiene testo, Carattere, schermata, linea

Descrizione generata automaticamente

Con questa query sono andando a calcolare il numero massimo di canzoni presenti in un disco.

Query 2:

Immagine che contiene testo, schermata

Descrizione generata automaticamente

Qui sono andando a calcolare il numero di serie del disco che contiene il numero maggiore di canzoni.

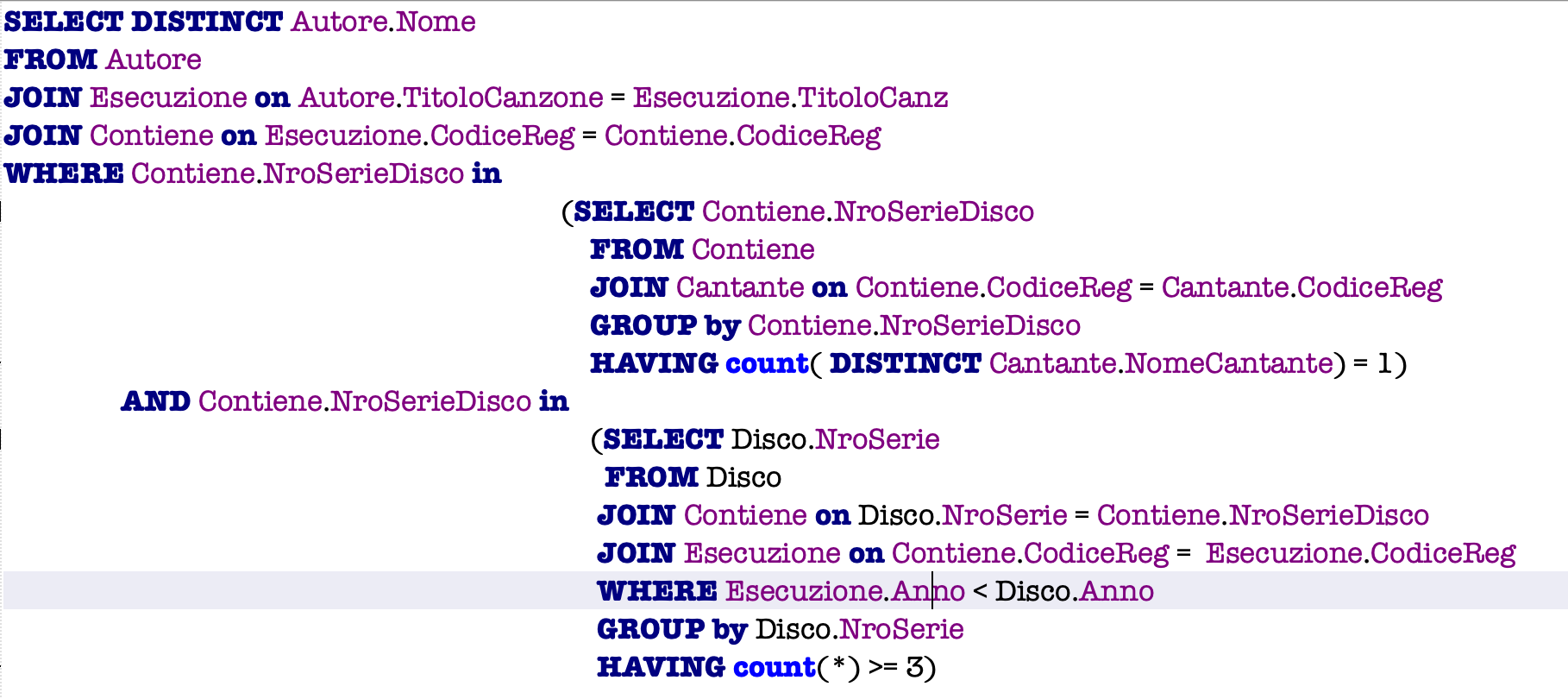
Query 3:

Immagine che contiene testo, Carattere, schermata, linea

Descrizione generata automaticamente

Vado a trovare il nome del cantante che è presente in quel disco. Uso il distinct per non avere doppioni dei nomi dei cantanti nel caso in cui uno di essi partecipi con più canzoni.

1. Gli autori solisti di “collezioni di successi” (dischi in cui tutte le canzoni sono di un solo cantante e in cui almeno tre registrazioni sono di anni precedenti la pubblicazione del disco)



DISCO(NroSerie, TitoloAlbum, Anno, Prezzo)

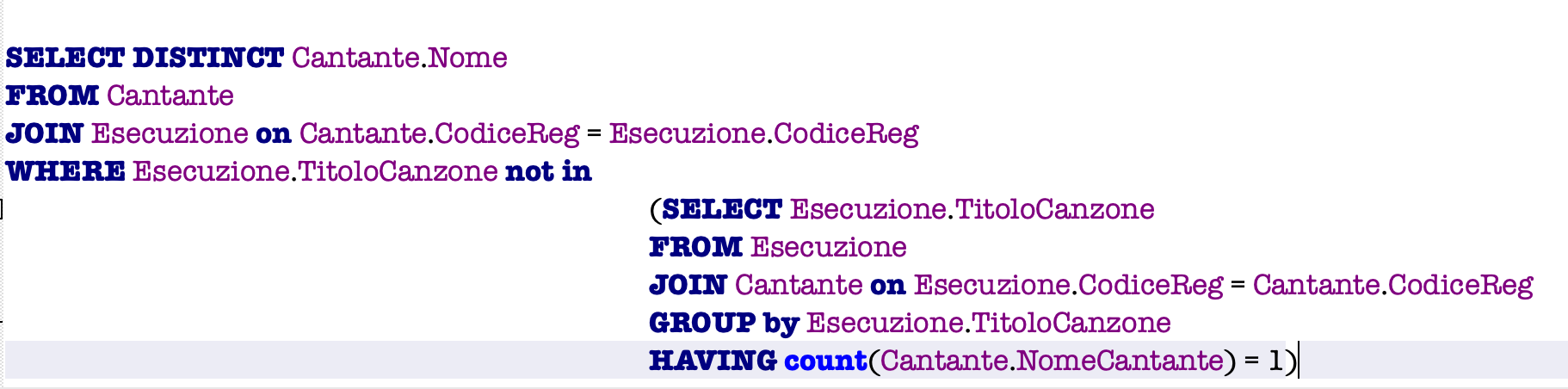
CONTIENE (NroSerieDisco, CodiceReg, NroProgr)

ESECUZIONE(CodiceReg, TitoloCanz, Anno)

AUTORE(Nome, TitoloCanzone)

CANTANTE(NomeCantante, CodiceReg)

1. I cantanti che non hanno mai registrato una canzone come solisti



1. I cantanti che non hanno mai inciso un disco in cui comparissimo come unici cantanti

NroSerie, CodiceReg, NomeCantante,NroProgr, TitoloAlbum, Anno, Prezzo

NroSerieDisco, CodiceReg, NomeCantante,

Esercizio 4.9

Dare una sequenza di comandi di aggiornamento che modifichi l’attriubuto Stipendio della tabella Impiegato, aumentando del 10% gli stipendisotto i 30 mila euro e diminuendo del 5 %gli stipendi sopra i 30 mila euro.

Update Impiegato

Set Stipendio = Stipendio \* 1.1

Where Stipendio <= 30

Update Impiegato

Set Stipendio = Stipendio \* 0,95

Where Stipendio > 30

Se Avessi un dipendente con stipendio iniziale uguale a 30 mila euro, esso subisce prima un aumento e successivamente un riabbasso.

Update Impiegato

Set Stipendio = Stipendio \* 0,95

Where Stipendio > 30

Update Impiegato

Set Stipendio = Stipendio \* 1.1

Where Stipendio <= 30

Se avessi un dipendente con stipendio iniziale uguale a 30 mila euro, prima verrebbero diminuiti gli stipendi superiori e successivamente aumentati quelli inferiori. Invertendo l’ordine c’è effettivamente un miglioramento? Perché per quegli stipendi sopra a 30 che vengono diminuiti e si ritrovano sotto i 30 adesso, non verrebbero comunque poi aumentati?

**ESERCIZI POLITECNICO**

Si consideri il seguente schema di base di dati, che descrive le vendite di bevande effettuate da una catena di punti vendita costituiti da soli distributori automatici.

Bevanda (Codice, Nome, Prezzo, QtaCL)

Vendita (CodDistributore, Data, Ora, CodBevanda, QtaZucchero, UsoChiavetta)

Distributore (Codice, CapienzaCL, DataRifornimento, OraRifornimento)

Le bevande sono “caffè espresso”, “caffè lungo”, “cioccolata”, …

*QtaCL* esprime, in centilitri, la quantità di bevanda.

L’attributo *QtaZucchero* specifica, su una scala da 0 a 5 la quantità di zucchero scelta durante l’acquisto

*UsoChiavetta* assume valore TRUE oppure FALSE a seconda che l’acquisto sia stato effettuato utilizzando la chiavetta o no

*CapienzaCL* esprime, in centilitri, la capienza totale di bevande del distributore.

* Trovare i codici delle bevande vendute il 15 luglio del 2016

SELECT DISTINCT Vendita.CodBevanda

FROM Vendita

WHERE Data = ‘15/07/2016’

* Trovare il codice del distributore e il codice della bevanda delle vendite effettuate nell’agosto 2016

SELECT DISTINCT CodDistributore, CodBevanda

FROM Vendita

WHERE Data >= ‘01/07/2016 and Data <= ‘31/07/2016’

* Trovare la bevanda più costosa

SELECT \*

FROM Bevanda

WHERE Prezzo = (SELECT max(Prezzo)

FROM Bevanda)

SELECT \*

FROM Bevanda

WHERE Prezzo >= ALL SELECT Prezzo -> un altro modo per farlo

FROM Bevanda

* Trovare le vendite fatte dai distributori successivamente al rifornimento

SELECT Vendita.\*

FROM Vendita JOIN Distributore

on Vendita.CodDistributore = Distributore.Codice

WHERE Vendita.Data > Distributore.DataRifornimento OR

(Vendita.Data = Distributore.DataRifornimento AND

Vendita.Ora > Distributore.OraRiFornimento)

* Trovare i distributori che non hanno mai venduto niente

SELECT \*

FROM Distributore

WHERE NOT EXISTS (

SELECT \*

FROM Vendita

WHERE Distributore.Codice = Vendita.CodDistributore);

* Trovare i distributori che non hanno fatto nessuna vendita successiva al loro rifornimento

SELECT \*

FROM Distributore

WHERE Distributore.Codice NOT In

( SELECT Vendita.CodiceDistributore

FROM Vendita

JOIN Distributore on Vendita.CodDistibutore = Distributore.Codice

WHERE Vendita.Data > Distibuore.DataRifornimento

OR (Vendita.Data = Distributore.DataRifornimento AND Vendita.Ora > Distributore.DataRifornimento)

* Trovare i distributori che non hanno mai venduto la bevanda di codice “*B1*”

SELECT \*

FROM Distributore

WHERE NOT EXISTS (SELECT \*

FROM Vendita

WHERE Distributore.Codice = Vendita.CodDistributore AND Vendita.CodBevanda = 'B1')

* Trovare i distributori che non hanno **mai** venduto “*tè al limone*”.