QUERY:

QUERY 1

**select count(distinct language.name) as Numero\_Lingue**

**from release join language**

**on release.language=language.id**

1. La query utilizza le release e le lingue per trovare il risultato.
2. In questa query é bastato un unico join tra release e language sull’attributo language di release(una chiave esterna) con l’identificatore di language (la sua chiave primaria), possibili valori nulli sono dati dall’attributo language di release.
3. In questa query non sono presenti ulteriori sottoproblemi.
4. La gestione dei duplicati é data da distinct mentre i valori nulli vengono scartati dal conteggio direttamente dal join (esso infatti esegue un “all implicito” di default, eliminando così tutte le ennuple con valore nullo nell’attributo interessato).
5. Non sono richieste due versioni di questa query.
6. Contare il numero di righe del risultato della seguente query:

**select distinct language.name from release join language on release.language=language.id**

La query riporta come risultato le lingue con cui sono scritte le release, contandole si scopre che il risultato corrisponde al conteggio effettuato dalla prima.

QUERY 2

**select distinct artist.name,**

**language.name**

**from release join language**

**on release.language=language.id**

**join artist\_credit**

**on artist\_credit.id=release.artist\_credit**

**join artist\_credit\_name**

**on artist\_credit.id=artist\_credit\_name.artist\_credit**

**join artist**

**on artist\_credit\_name.artist=artist.id**

**where language.name='English'**

**1)** Nel risultato vengono richiesti gli artisti e la lingua (attributo name della tabella artist e attributo name della tabella language)della canzone che cantano. (si è scelto Inglese perché mancava Italiano).

**2)** Per questa query si è partiti con un join tra release e language sull’attributo language della tabella release e sull’attributo id della tabella language.Il join è tra la chiave esterna Language della tabella release e la chiave primaria id della tabella Language. Viene poi fatto un altro join è tra l’attributo id della tabella artist\_credit e l’attributo artist\_credit nella tabella release. Un altro join è fatto tra la chiave esterna Artist\_credit di release e la chiave primaria id nella tabella artist\_credit. Il prossimo join è tra l’attributo id della tabella artist\_credit e l’attributo artist\_credit in artist\_credit\_name.Il join è quindi fatto sulle due chiavi primarie delle due tabelle. L’ ultimo è tra l’attributo id della tabella artist e l’attributo artist in artist\_credit\_name.Il join è fatto sulle due chiavi primarie delle due tabelle.

**3)** Nella query non sono presenti sotto problemi.

**4)** Per la gestione dei duplicati si è utilizzato il distinct, i valori nulli non vengono considerati perché viene scelta solo la lingua Inglese. Sull’attributo Language di Release sono presenti valori nulli ma non interessano al risultato.

**5)** Non sono richieste due versioni della query.

**6)**

**select distinct artist.name,language.name**

**from release join language**

**on release.language=language.id**

**join artist\_credit**

**on artist\_credit.id=release.artist\_credit**

**join artist\_credit\_name**

**on artist\_credit.id=artist\_credit\_name.artist\_credit**

**join artist**

**on artist\_credit\_name.artist=artist.id**

**where language.name not in**

**(**

**select distinct language.name**

**from release join language**

**on release.language=language.id**

**join artist\_credit**

**on artist\_credit.id=release.artist\_credit**

**join artist\_credit\_name**

**on artist\_credit.id=artist\_credit\_name.artist\_credit**

**join artist**

**on artist\_credit\_name.artist=artist.id**

**where language.name='English'**

**)**

Con la query di verifica prendo tutti gli artisti che non hanno cantato canzoni in Inglese e li cerco dove ci sono tutti quelli che hanno invece cantato canzoni in Inglese e non trovo risultati di artisti che hanno cantato canzoni in inglese quindi significa che la query è corretta.

QUERY 3

**select release.name**

**from release**

**where release.language is null**

1. La query utilizza solo release per il calcolo del risultato.
2. La query non presenta nessun join con altre tabelle dello schema e la chiave primaria di release (il suo identificatore) non viene interessata dalla query. L’attributo language di release è una chiave esterna, il quale potrebbe anche contenere possibili valori nulli.
3. Questa query non richiede ulteriori sottoproblemi per essere risolta.
4. In questa query vengono considerati i valori nulli dell’attributo language di release che servono per il calcolo del risultato finale (nello schema è segnato col grassetto).
5. Non sono richieste ulteriori versioni di questa query.

**select release.id, release.name**

**from release**

**where release.language is not null**

Con questa query contraria vedo che non ci sono nomi di release che hanno l’attributo language a null, quindi concludo che la query è giusta.

QUERY 4

**select release.name**

**from release**

**where release.name like '%Love%'**

**1)** Nella query viene richiesto solo il nome della release (attributo name della tabella release) che contiene la parola Love.

**2)** Non si sono usati join per questa query.

**3)** Non esistono sotto problemi per questa query.

**4)** I valori nulli non sono presenti sull’attributo name di release.

**5)** Non sono richieste due versioni della query.

**6)**

**select release.name**

**from release**

**where release.name not in**

**(**

**select release.name**

**from release**

**where release.name like '%Love%'**

**)**

La query permette di verificare il risultato perché sto cercando le release che non hanno la parola Love e, cercandole dove effettivamente c’è la parola Love, non trovo risultati che contengano la parola Love. Quindi significa che la query è corretta.

QUERY 5

**select artist\_alias.name, artist.name, artist\_alias\_type.name**

**from artist\_alias join artist**

**on artist.id=artist\_alias.artist**

**join artist\_alias\_type**

**on artist\_alias.type=artist\_alias\_type.id**

**where artist.name='Prince'**

1. la query si serve di artist, artist\_alias e artist\_alias\_type per il calcolo del risultato.
2. la query presenta due join: uno connette artist con artist\_alias sulla chiave primaria del primo (il suo identificatore) con l’attributo artist dell’ultimo (una chiave esterna), l’altro invece connette artist\_alias con artist\_alias\_type rispettivamente sulla chiave esterna type con la chiave primaria (il suo identificatore) di quest’ultima. Possibili valori nulli possono essere presenti negli attributi artist e type di artist\_alias.
3. la query non presenta ulteriori sottoproblemi da trattare.
4. Nella query vengono utilizzati attributi che potrebbero essere nulli, tuttavia, oltre a non interessare in alcun modo il risultato, esse vengono escluse direttamente dai join.
5. la query non richiede due versioni della stessa.

**select artist\_alias.name, artist.name, artist\_alias\_type.name**

**from artist\_alias join artist**

**on artist.id=artist\_alias.artist**

**join artist\_alias\_type**

**on artist\_alias.type=artist\_alias\_type.id**

**where artist.name='Prince'**

**intersect**

**select artist\_alias.name, artist.name, artist\_alias\_type.name**

**from artist\_alias join artist**

**on artist.id=artist\_alias.artist**

**join artist\_alias\_type**

**on artist\_alias.type=artist\_alias\_type.id**

**where artist.name<>'Prince'**

La query verifica il risultato perché l’intersezione tra gli pseudonimi di Prince e quelli che non lo sono è nulla, questo mi permette di dire che la query richiesta dà i risultati esatti.

QUERY 6

**select distinct artist.name,release.name**

**from artist join artist\_type**

**on artist.type=artist\_type.id**

**join artist\_credit\_name**

**on artist\_credit\_name.artist=artist.id**

**join artist\_credit**

**on artist\_credit.id=artist\_credit\_name.artist\_credit**

**join release**

**on artist\_credit.id=release.artist\_credit**

**join area**

**on artist.area=area.id**

**where artist\_type.name='Group'**

**and artist.ended=false**

**and area.name='United Kingdom'**

**order by artist.name, release.name**

**1)** Nella query è richiesto il nome del gruppo (attributo name della tabella artist) e il nome della release (attributo name della tabella release) di gruppi inglesi ancora in attività. Il tutto deve essere ordinato per nome della release e nome dell’artista.

**2)** Viene fatto un join tra la chiave esterna type della tabella artist e la chiave primaria id della tabella artist\_type. Il prossimo join è tra la chiave primaria id della tabella artist e la chiave esterna artist della tabella artist\_credit\_name. Il successivo è fatto sulle chiavi primarie delle tabelle artist\_credit e artist\_credit\_name. Poi c’è il join tra la chiave primaria di artist\_credit e la chiave esterna della tabella release artist\_credit. L’ultimo join è tra la chiave primaria della tabella area e la chiave esterna area della tabella artist.

**3)** Non sono presenti sotto problemi.

**4)** Per eliminare i duplicati si è usato il distinct, i valori nulli non si sono considerati in quanto non presenti sugli attributi.

**5)** Non sono richieste due versioni della query.

**6)**

**select distinct artist.name,release.name**

**from artist join artist\_type**

**on artist.type=artist\_type.id**

**join artist\_credit\_name**

**on artist\_credit\_name.artist=artist.id**

**join artist\_credit**

**on artist\_credit.id=artist\_credit\_name.artist\_credit**

**join release**

**on artist\_credit.id=release.artist\_credit**

**join area**

**on artist.area=area.id**

**where artist\_type.name='Group'**

**and area.name='United Kingdom'**

**and artist.name not in**

**(**

**select distinct artist.name**

**from artist join artist\_type**

**on artist.type=artist\_type.id**

**join artist\_credit\_name**

**on artist\_credit\_name.artist=artist.id**

**join artist\_credit**

**on artist\_credit.id=artist\_credit\_name.artist\_credit**

**join release**

**on artist\_credit.id=release.artist\_credit**

**join area**

**on artist.area=area.id**

**where artist\_type.name='Group'**

**and artist.ended=true**

**and area.name='United Kingdom'**

**order by artist.name**

**)**

La query permette di verificare il risultato perché, cercando i gruppi nell’insieme di quelli che hanno già concluso la loro attività, trovo tutti quelli che non l’hanno ancora conclusa. Posso così dire che la query è giusta.

QUERY 7

**select distinct release.name,**

**artist.name,**

**artist\_credit.name**

**from artist join artist\_credit\_name**

**on artist.id=artist\_credit\_name.artist**

**join artist\_credit**

**on artist\_credit.id=artist\_credit\_name.artist\_credit**

**join release**

**on release.artist\_credit=artist\_credit.id**

**where artist\_credit.name<>artist.name**

1. la query utilizza artist, artist\_credit, artist\_credit\_name e release per il calcolo del risultato.
2. Nella query vi sono tre join:

* join tra artist e artist\_credit\_name sulla chiave primaria del primo (l’identificatore) con la chiave esterna artist del secondo.
* join tra artist\_credit\_name e artist\_credit in cui si mettono in relazione la chiave esterna artist\_credit del primo con la chiave primaria del secondo (il suo identificatore).
* join tra artist\_credit e release dove l’identificatore del primo (la sua chiave primaria) viene messo in relazione con la chiave esterna artist\_credit del secondo.

3) La query non presenta sotto problemi da risolvere.

4) La gestione dei duplicati é data da distinct mentre i valori nulli, oltre a non essere necessari per il risultato finale, vengono direttamente esclusi dai join.

5) Non sono richieste altre versioni di questa query.

6)

**select distinct release.name,**

**artist.name,**

**artist\_credit.name**

**from artist join artist\_credit\_name**

**on artist.id=artist\_credit\_name.artist**

**join artist\_credit**

**on artist\_credit.id=artist\_credit\_name.artist\_credit**

**join release**

**on release.artist\_credit=artist\_credit.id**

**where artist\_credit.name<>artist.name**

**intersect**

**select distinct release.name,**

**artist.name,**

**artist\_credit.name**

**from artist join artist\_credit\_name**

**on artist.id=artist\_credit\_name.artist**

**join artist\_credit**

**on artist\_credit.id=artist\_credit\_name.artist\_credit**

**join release**

**on release.artist\_credit=artist\_credit.id**

**where artist\_credit.name = artist.name**

La query verifica il risultato perché l’intersezione tra le release in cui il nome dell’artista è diverso dal nome accreditato nella release e tra le release in cui invece è uguale è nulla, questo mi permette di dire che la query richiesta dà i risultati esatti.

QUERY 8

**select artist.name, count(release.name)**

**from artist join artist\_credit\_name**

**on artist.id=artist\_credit\_name.artist**

**join artist\_credit**

**on artist\_credit.id=artist\_credit\_name.artist\_credit**

**join release**

**on artist\_credit.id=release.artist\_credit**

**group by artist.name**

**having count(release.name)<3**

**1)** Nella query viene richiesto il nome dell’artista (attributo name della tabella artist) e il numero delle sue release.

**2)** Viene fatto il join tra la chiave primaria id della tabella artist con la chiave esterna artist della tabella artist\_credit\_name. L’altro join è tra le chiavi primarie di artist\_credit e artist\_credit\_name. L’ultimo join è tra la chiave primaria di artist\_credit e la chiave esterna artist\_credit della tabella release. Non sono presenti valori nulli sugli attributi su cui si è fatto il join.

**3)** Non esistono sotto problemi.

**4)** Non sono presenti duplicati quindi non si è usato il distinct, i valori nulli non sono stati presi in considerazione.

**5)** Non sono richieste due versioni della query.

**6)**

**select artist.name, count(release.name)**

**from artist join artist\_credit\_name**

**on artist.id=artist\_credit\_name.artist**

**join artist\_credit**

**on artist\_credit.id=artist\_credit\_name.artist\_credit**

**join release**

**on artist\_credit.id=release.artist\_credit**

**where artist.name not in**

**(**

**select artist.name**

**from artist join artist\_credit\_name**

**on artist.id=artist\_credit\_name.artist**

**join artist\_credit**

**on artist\_credit.id=artist\_credit\_name.artist\_credit**

**join release**

**on artist\_credit.id=release.artist\_credit**

**group by artist.name**

**having count(release.name)<3**

**)**

**group by artist.name**

Con questa query di controllo cerco tutti gli artisti con un numero di release superiori a 3 e non ne trovo nessuno che ne abbia un numero inferiore a 3. Si deduce quindi che la query sia giusta.

QUERY 9 (versione 1)

**select artist.name,**

**recording.name,**

**recording.length/60000**

**from artist join gender**

**on artist.gender=gender.id**

**join artist\_credit\_name**

**on artist\_credit\_name.artist=artist.id**

**join artist\_credit**

**on artist\_credit\_name.artist\_credit=artist\_credit.id**

**join recording**

**on recording.artist\_credit=artist\_credit.id**

**where gender.name='Female'**

**and recording.length is not null**

**and recording.length=**

**(**

**select max(recording.length)**

**from artist join gender**

**on artist.gender=gender.id**

**join artist\_credit\_name**

**on artist\_credit\_name.artist=artist.id**

**join artist\_credit**

**on artist\_credit\_name.artist\_credit=artist\_credit.id**

**join recording**

**on recording.artist\_credit=artist\_credit.id**

**where gender.name='Female'**

**and recording.length is not null**

**)**

1. la query utilizza artist, artist\_credit, artist\_credit\_name e recording per il calcolo del risultato. Il calcolo della registrazione più lunga é dato da una sottoquery in cui si utilizzano le stesse entità per il calcolo.
2. Nella query vi sono tre join:

* join tra artist e artist\_credit\_name sulla chiave primaria del primo (l’identificatore) con la chiave esterna artist del secondo.
* join tra artist\_credit\_name e artist\_credit in cui si mettono in relazione la chiave esterna artist\_credit del primo con la chiave primaria del secondo (il suo identificatore).
* join tra artist\_credit e recording dove l’identificatore del primo (la sua chiave primaria) viene messo in relazione con la chiave esterna artist\_credit del secondo.

Possibili valori nulli sull’attributo recording.length, ma non si sono considerati perché inutili al fine del risultato e quindi si sono esclusi.

3) L’unico sottoproblema di questa query é il calcolo della registrazione più grande tra tutte. Essa confronta tutte le lunghezze delle registrazioni d’ogni artista donna e, attraverso l’operatore aggregato max, calcola il risultato (é formata da un solo valore). Il risultato della sottoquery verrà poi utilizzato in una clausola where in cui lo si confronta con tutte le lunghezze per verificare qual é quella più alta.

4)la query non utilizza particolari assunzioni poiché il risultato della query é formato da una sola riga e i valori nulli vengono esclusi dai join.

5)E’ richiesta una versione alternativa della query in cui non bisogna utilizzare l’operatore aggregato max (vedere in seguito).

6)

**select artist.name,**

**recording.name,**

**recording.length/60000**

**from artist join gender**

**on artist.gender=gender.id**

**join artist\_credit\_name**

**on artist\_credit\_name.artist=artist.id**

**join artist\_credit**

**on artist\_credit\_name.artist\_credit=artist\_credit.id**

**join recording**

**on recording.artist\_credit=artist\_credit.id**

**where gender.name='Female'**

**and recording.length is not null**

**order by recording.length desc**

La query permette di verificare il risultato perché si è ordinato in modo decrescente le registrazioni degli artisti donna e la più lunga(il primo risultato) risulta essere proprio quella ottenuta nella query richiesta quindi posso concludere sia giusta. Si sono esclusi i valori nulli perché altrimenti li avrebbe messi come primi risultati ma essi non sono utili al fine del risultato.

QUERY 9 (versione 2)

**select \* from**

**(**

**select artist.name,recording.name,recording.length/60000 as length**

**from artist join gender**

**on artist.gender=gender.id**

**join artist\_credit\_name**

**on artist\_credit\_name.artist=artist.id**

**join artist\_credit**

**on artist\_credit\_name.artist\_credit=artist\_credit.id**

**join recording**

**on recording.artist\_credit=artist\_credit.id**

**where gender.name='Female'**

**and recording.length is not null**

**)query**

**where query.length>= all**

**(**

**select recording.length/60000 as length**

**from artist join gender**

**on artist.gender=gender.id**

**join artist\_credit\_name**

**on artist\_credit\_name.artist=artist.id**

**join artist\_credit**

**on artist\_credit\_name.artist\_credit=artist\_credit.id**

**join recording**

**on recording.artist\_credit=artist\_credit.id**

**where gender.name='Female'**

**and recording.length is not null**

**)**

1. la query utilizza artist, artist\_credit, artist\_credit\_name e recording per il calcolo del risultato. Il calcolo della registrazione più lunga é dato da una sottoquery in cui si utilizzano le stesse entità per il calcolo.
2. Nella query vi sono tre join:

* join tra artist e artist\_credit\_name sulla chiave primaria del primo (l’identificatore) con la chiave esterna artist del secondo.
* join tra artist\_credit\_name e artist\_credit in cui si mettono in relazione la chiave esterna artist\_credit del primo con la chiave primaria del secondo (il suo identificatore).
* join tra artist\_credit e recording dove l’identificatore del primo (la sua chiave primaria) viene messo in relazione con la chiave esterna artist\_credit del secondo.

Possibili valori nulli possono essere presenti tra gli attributi segnati come chiave esterna.

3) L’unico sottoproblema di questa query é il calcolo della registrazione più grande tra tutte. Essa confronta tutte le lunghezze delle registrazioni d’ogni artista donna e, attraverso l’operatore aggregato max, calcola il risultato (é formata da un solo valore). Il risultato della sottoquery verrà poi utilizzato in una clausola where in cui lo si confronta con tutte le lunghezze per verificare qual é quella più alta.

4)la query non utilizza particolari assunzioni poiché il risultato della query é formato da una sola riga e i valori nulli vengono esclusi dai join.

5)E’ richiesta un’ulteriore versione della query in cui bisogna utilizzare l’operatore aggregato max (vedere sopra).

6)La modalità di verifica del risultato è le stessa della versione 1.

QUERY 10 ( versione 1)

**select language.name, count(release.name)**

**from language left join release**

**on language.id=release.language**

**group by language.name**

**having count(release.name)=0**

**order by language.name**

**1)** Il risultato chiede il nome della lingua (attributo name della tabella Language) e il numero delle release.

**2)** Viene fatto un left join tra la chiave primaria della tabella Language e la chiave esterna della tabella release. Questo viene fatto per considerare i valori nulli che ha l’attributo language della tabella release. Quindi, grazie a ciò, mette count = 0 alle lingue cui non corrisponde nessuna release.

**3)** Non sono presenti sotto problemi.

**4)** Non è necessario usare distinct per i duplicati e per quanto riguarga i valori nulli, ho già spiegato al punto precedente.

**5)** E’ richiesta una seconda versione della query.

**6)**

**select language.name,count(release.name)**

**from language join release**

**on language.id=release.language**

**group by language.name**

**having count(release.name) <>0**

**order by language.name**

Questa query cerca tutte le lingue cui corrisponde almeno una release e non compaiono le lingue cui non corrisponde nessuna release. Si conclude che la query richiesta è giusta.

QUERY 10 (versione 2)

**select language.name,**

**0 as count**

**from language**

**where language.name not in**

**(**

**select distinct language.name**

**from language join release**

**on language.id = release.language**

**)**

**group by language.name**

**order by language.name**

**1)** Il risultato chiede il nome della lingua (attributo name della tabella Language) e il numero delle release.

**2)** Il join viene fatto nel sotto problema tra la chiave primaria della tabella language e la chiave esterna language della tabella release.

**3)** E’ presente un sotto problema in cui vengono considerate tutte le lingue cui corrisponde almeno una release. Si cercano poi qui tutte le lingue che non sono contenute in questo insieme e si mette a queste il count=0, perché sicuramente queste lingue non avranno release.

**4)** Duplicati e valori nulli non vengono considerati in questa query.

**5)** Si è già fatta una prima versione della query.

**6)**

Per controllare la query ho già i risultati della prima versione, che coincidono, la verifica è già stata fatta nella prima versione quindi la query è giusta.

QUERY 11 (versione 1)

**select artist.name,recording.name,recording.length**

**from artist join artist\_credit\_name on artist\_credit\_name.artist=artist.id**

**join artist\_credit on artist\_credit\_name.artist\_credit=artist\_credit.id**

**join recording on artist\_credit.id=recording.artist\_credit**

**join gender on artist.gender=gender.id**

**where gender.name='Male'**

**and recording.length is not null**

**and recording.length =(**

**select max(query.length) from(**

**select recording.name,recording.length**

**from artist join artist\_credit\_name on artist\_credit\_name.artist=artist.id**

**join artist\_credit on artist\_credit\_name.artist\_credit=artist\_credit.id**

**join recording on artist\_credit.id=recording.artist\_credit**

**join gender on artist.gender=gender.id**

**where gender.name='Male'**

**and recording.length is not null**

**except**

**select recording.name,recording.length**

**from artist join artist\_credit\_name on artist\_credit\_name.artist=artist.id**

**join artist\_credit on artist\_credit\_name.artist\_credit=artist\_credit.id**

**join recording on artist\_credit.id=recording.artist\_credit**

**join gender on artist.gender=gender.id**

**where gender.name='Male'**

**and recording.length is not null**

**and recording.length=(**

**select max(recording.length)**

**from artist join artist\_credit\_name on artist\_credit\_name.artist=artist.id**

**join artist\_credit on artist\_credit\_name.artist\_credit=artist\_credit.id**

**join recording on artist\_credit.id=recording.artist\_credit**

**join gender on artist.gender=gender.id**

**where gender.name='Male'**

**and recording.length is not null**

**)**

**) query**

**)**

1. La query prende il risultato dalla tabella risultante di una sottoquery, per farla vengono utilizzati artist, artist\_credit, artist\_credit\_name e recording.
2. la tabella utilizzata dalla query é il risultato della sottrazione insiemistica tra due ulteriori query, entrambe utilizzano i seguenti join:

* Join tra artist e artist\_credit\_name in cui si associa la chiave primaria del primo (in suo identificatore) con la chiave esterna artist del secondo.
* Join tra artist\_credit e artist\_credit\_name n cui si associa la chiave primaria del primo (in suo identificatore) con la chiave esterna artist\_credit del secondo.
* Join tra artist\_credit e recording in cui si mettono in relazione l’idenficatore del primo (la sua chiave primaria) con la chiave esterna artist\_credit del secondo.

Possibili valori nulli sono presenti sull’attributo recording.length, ma non vengono presi in considerazione per il risultato perchè esclusi.

3) Nella query sono presenti i seguenti sottoproblemi: il primo riguarda il calcolo del di tutte le registrazioni di artisti uomini senza la registrazione più lunga, essa infatti é stata tolta attraverso l’utilizzo d’un operatore except e una sottoquery che calcola la registrazione di lunghezza massima. Quest’ultima sottoquery funziona nel seguente modo: un ulteriore sottoquery calcola la lunghezza massima attraverso l’operatore aggregato max e, per ottenere la registrazione corrispondente, il risultato viene confrontato con tutte le lunghezze delle registrazioni.

Tutti i sottoproblemi utilizzano i join trattati nel punto 2.

4) la query, dato che il suo risultato é formato da una sola riga, non ha bisogno d’alcun controllo dei duplicato e, in più, i valori nulli vengono esclusi direttamente dai join.

5) é prevista un’ulteriore versione della query in cui non bisogna utilizzare le viste (vedere in seguito).

6)

**select artist.name,recording.name,recording.length**

**from artist join artist\_credit\_name on artist\_credit\_name.artist=artist.id**

**join artist\_credit on artist\_credit\_name.artist\_credit=artist\_credit.id**

**join recording on artist\_credit.id=recording.artist\_credit**

**join gender on artist.gender=gender.id**

**where gender.name='Male'**

**and recording.length is not null**

**order by recording.length desc**

La query verifica il risultato perchè vengono trovate tutte le registrazioni degli artisti uomini e vengono ordinati in modo decrescente, la seconda in alto coincide con il risultato ottenuto dalla query richiesta quindi trovo la seconda registrazione più lunga di un artista uomo. Questo mi permette di dire che la query richiesta è corretta.

QUERY 11 (versione 2)

**create view MaxRegUomo as // crea vista con la registrazione più lunga di un uomo**

**select max(recording.length)**

**from recording join artist**

**on artist.id=recording.artist\_credit**

**join gender on artist.gender=gender.id**

**join artist\_credit\_name on artist\_credit\_name.artist= artist.id**

**join artist\_credit on artist\_credit.id = artist\_credit\_name.artist\_credit**

**where gender.name = 'Male'**

**create view AllReg as //trova tutte le registrazioni di un uomo**

**select artist\_credit.name,recording.name as recName,recording.length as length**

**from recording join artist**

**on artist.id=recording.artist\_credit**

**join gender on artist.gender=gender.id**

**join artist\_credit\_name on artist\_credit\_name.artist= artist.id**

**join artist\_credit on artist\_credit.id = artist\_credit\_name.artist\_credit**

**where gender.name= 'Male'**

**and recording.length is not null**

**select AllReg.name,AllReg.recName,length**

**from AllReg**

**where length=**

**(**

**select max(query.length)**

**from**

**(**

**select AllReg.name,AllReg.length**

**from AllReg**

**except**

**select AllReg.name, AllReg.length**

**from AllReg**

**where AllReg.length=**

**(**

**select \* from MaxRegMen**

**)**

**)query**

**)**

1. la query prende informazioni dal risultato di una sottoquery, esse utilizzano delle viste per il calcolo del risultato. Le viste utilizzano le stesse tabelle citate nel primo punto delle QUERY 9(versione 1).
2. tutti i join fatti nelle viste sono trattati nell’altra versione della query.
3. Tutti i sottoproblemi della versione 1 della query sono stati inglobati nella viste, l’unico sotto problema esistente riguarda il calcolo della registrazione di lunghezza massima che verrà poi confrontata con tutte le righe di una delle viste.
4. Vedere il punto 4 della versione 1 della query.
5. é richiesta una versione della query in cui non é richiesto l’utilizzo delle viste.
6. La modalità per verificare la correttezza del risultato è analoga a quella della versione 1.

QUERY 12 (versione 1)

**select area.name, sum(recording.length)/60000**

**from area join area\_type**

**on area.type=area\_type.id**

**join artist**

**on artist.area=area.id**

**join artist\_credit\_name**

**on artist\_credit\_name.artist=artist.id**

**join artist\_credit**

**on artist\_credit\_name.artist\_credit=artist\_credit.id**

**join recording**

**on recording.artist\_credit=artist\_credit.id**

**where area\_type.name='Country'**

**group by area.name**

**having sum(recording.length) is not null**

**union**

**select area.name, 0 as length**

**from area join area\_type**

**on area.type=area\_type.id**

**join artist**

**on artist.area=area.id**

**join artist\_credit\_name**

**on artist\_credit\_name.artist=artist.id**

**join artist\_credit**

**on artist\_credit\_name.artist\_credit=artist\_credit.id**

**join recording**

**on recording.artist\_credit=artist\_credit.id**

**where area\_type.name='Country'**

**group by area.name**

**having sum(recording.length) is null**

**1)** Il risultato chiede il nome dello stato (attributo name della tabella area) e la somma delle lunghezze delle registrazioni degli artisti di quello Stato.

**2)** Il primo join è tra la chiave primaria della tabella area\_type e la chiave esterna type della tabella area, il successivo è tra la chiave primaria della tabella area e la chiave esterna area della tabella artist. Viene poi fatto il join tra la chiave primaria id della tabella artist con la chiave esterna artist della tabella artist\_credit\_name. L’altro join è tra le chiavi primarie di artist\_credit e artist\_credit\_name. L’ultimo join è tra la chiave esterna artist\_credit della tabella recording e la chiave primaria della tabella artist\_credit.

Nella query viene utilizzato l’operatore insiemistico union per mettere insieme i risultati degli stati che hanno 0 come somma delle registrazioni con quelli degli stati con somma delle registrazioni maggiore di 0.

**3)** Non sono presenti sotto problemi.

**4)** Valori duplicati non sono presenti, per quanto riguarda i valori nulli si tengono in considerazione nella seconda parte della query e vengono considerati zero, cioè viene messo il valore 0 dove la somma delle registrazioni è null.

**5)** La seconda versione della query è realizzata con le viste.

**6)**

**select area.name, sum(recording.length)/60000**

**from area join area\_type**

**on area.type=area\_type.id**

**join artist**

**on artist.area=area.id**

**join artist\_credit\_name**

**on artist\_credit\_name.artist=artist.id**

**join artist\_credit**

**on artist\_credit\_name.artist\_credit=artist\_credit.id**

**join recording**

**on recording.artist\_credit=artist\_credit.id**

**where area\_type.name!='Country'**

**group by area.name**

**having sum(recording.length) is null**

**and area.name in**

**(**

**select area.name**

**from area join area\_type**

**on area.type=area\_type.id**

**join artist**

**on artist.area=area.id**

**join artist\_credit\_name**

**on artist\_credit\_name.artist=artist.id**

**join artist\_credit**

**on artist\_credit\_name.artist\_credit=artist\_credit.id**

**join recording**

**on recording.artist\_credit=artist\_credit.id**

**where area\_type.name='Country'**

**group by area.name**

**having sum(recording.length) is null**

**)**

Con questa query si verifica che non ci siano aree che non sono Stati in cui ci sia somma registrazioni=0.

**select area.name, sum(recording.length)/60000**

**from area join area\_type**

**on area.type=area\_type.id**

**join artist**

**on artist.area=area.id**

**join artist\_credit\_name**

**on artist\_credit\_name.artist=artist.id**

**join artist\_credit**

**on artist\_credit\_name.artist\_credit=artist\_credit.id**

**join recording**

**on recording.artist\_credit=artist\_credit.id**

**where area\_type.name!='Country'**

**group by area.name**

**having sum(recording.length) is not null**

**and area.name in**

**(**

**select area.name**

**from area join area\_type**

**on area.type=area\_type.id**

**join artist**

**on artist.area=area.id**

**join artist\_credit\_name**

**on artist\_credit\_name.artist=artist.id**

**join artist\_credit**

**on artist\_credit\_name.artist\_credit=artist\_credit.id**

**join recording**

**on recording.artist\_credit=artist\_credit.id**

**where area\_type.name='Country'**

**group by area.name**

**having sum(recording.length) is not null**

**)**

Con questa query si verifica che non ci siano aree che non sono Stati in cui ci sia somma registrazioni>0.

La verifica contemporanea di queste due query permette di dire che la query di partenza è corretta.

QUERY 12 (versione 2)

**create view RegNotNull as**

**select area.name, sum(recording.length)/60000**

**from area join area\_type**

**on area.type=area\_type.id**

**join artist**

**on artist.area=area.id**

**join artist\_credit\_name**

**on artist\_credit\_name.artist=artist.id**

**join artist\_credit**

**on artist\_credit\_name.artist\_credit=artist\_credit.id**

**join recording**

**on recording.artist\_credit=artist\_credit.id**

**where area\_type.name='Country'**

**group by area.name**

**having sum(recording.length) is not null**

**create view RegNull as**

**select area.name, 0 as length**

**from area join area\_type**

**on area.type=area\_type.id**

**join artist**

**on artist.area=area.id**

**join artist\_credit\_name**

**on artist\_credit\_name.artist=artist.id**

**join artist\_credit**

**on artist\_credit\_name.artist\_credit=artist\_credit.id**

**join recording**

**on recording.artist\_credit=artist\_credit.id**

**where area\_type.name='Country'**

**group by area.name**

**having sum(recording.length) is null**

**select \* from**

**RegNotNull**

**union**

**select \* from**

**RegNull**

**1)** I risultati della query sono tutti gli attributi della vista RegNotNull e di quella RegNull.

**2)** I join sono i medesimi della query senza viste.

**3)** Non sono presenti sotto problemi. Sono state create 2 viste, una con gli Stati con somma registrazioni = 0, una con somma registrazioni > 0.

**4)** Il discorso sui duplicati e sui valori nulli vale lo stesso fatto per la versione 1.

**5)** E’ stata fatta una prima versione della query.

**6)** La query è costruita sulla base di quella della versione 1, quindi i controlli fatti per la precedente valgono anche per questa.

QUERY 13 (versione 1)

**select artist.name, area.name, count(release.name)**

**from artist join area**

**on area.id=artist.area**

**join artist\_credit\_name**

**on artist\_credit\_name.artist=artist.id**

**join artist\_credit**

**on artist\_credit\_name.artist\_credit=artist\_credit.id**

**join release**

**on release.artist\_credit=artist\_credit.id**

**where area.name='United Kingdom'**

**group by artist.name, area.name**

**having count(release.name)>=10**

1. la query si serve delle tabelle artist, artist\_credit\_name, artist\_credit, release e area per calcolare il risultato.
2. Nella query sono presenti i seguenti join:

* join tra artist e artist\_credit\_name in cui l’identificatore e chiave primaria di artist viene messo in relazione con la chiave esterna artist di artist\_credit\_name.
* join tra artist\_credit\_name e artist\_credit dove vengono associati la chiave esterna artist\_credit di artist\_credit\_name con l’identificatore e chiave primaria di artist\_credit.
* join tra artist\_credit e release dove la chiave primaria di artist\_credit( il suo identificatore) viene messo il relazione con la chiave esterna artist\_credit di release.
* join tra artist e area in cui la chaive esterna area di artist viene legata all’identificatore e chiave primaria di area.

Possibili valori nulli potrebbero essere presenti negli attributi segnati come chiave primaria.

3) In questa query non sono presenti sottoproblemi da trattare ulteriormente.

4) La query non richiede la gestione dei duplicati e non é necessario considerare valori nulli perché vengono tolti direttamente dai join.

5)é richiesta una seconda versione della query in cui si devono utilizzare le viste.

6)

**select artist.name, area.name, count(release.name)**

**from artist join area**

**on area.id=artist.area**

**join artist\_credit\_name**

**on artist\_credit\_name.artist=artist.id**

**join artist\_credit**

**on artist\_credit\_name.artist\_credit=artist\_credit.id**

**join release**

**on release.artist\_credit=artist\_credit.id**

**where area.name='United Kingdom'**

**group by artist.name, area.name**

**having count(release.name)>=10**

**intersect**

**select artist.name, area.name, count(release.name)**

**from artist join area**

**on area.id=artist.area**

**join artist\_credit\_name**

**on artist\_credit\_name.artist=artist.id**

**join artist\_credit**

**on artist\_credit\_name.artist\_credit=artist\_credit.id**

**join release**

**on release.artist\_credit=artist\_credit.id**

**where area.name='United Kingdom'**

**group by artist.name, area.name**

**having count(release.name)<10**

Questa query permette di verificare il risultato perchè l’intersezione tra gli artisti britannici che hanno pubblicato almeno 10 release e quelli che ne hanno pubblicati meno è nulla, perciò posso dire che la query richiesta è corretta.

QUERY 13 (versione 2)

**create view britainRelease as**

**select artist.name as artName, area.name as areaName, release.name as relName**

**from artist join area**

**on area.id=artist.area**

**join artist\_credit\_name**

**on artist\_credit\_name.artist=artist.id**

**join artist\_credit**

**on artist\_credit\_name.artist\_credit=artist\_credit.id**

**join release**

**on release.artist\_credit=artist\_credit.id**

**where area.name='United Kingdom'**

**select artName,areaName,count(relName)**

**from britainRelease**

**group by artName, areaName**

**having count(relName)>=10**

1. La query prende le informazioni da una vista che utilizzata le stesse tabelle indicate nella versione 1 (punto 1).
2. Tutti i join del caso sono trattati nel precedente punto 2.
3. L’unico sottoproblema presente in questa query riguarda la vista, infatti in essa vengono raggruppate tutte le tabelle citate nel punto 1, join compresi.
4. tutte le assunzioni sono già state trattate nel punto 4 precedente.
5. é richiesta una seconda versione di questa query in cui non si devono utilizzare le viste.
6. La modalità di verifica del risultato è la stessa usata nella versione 1 perchè la versione con le viste è costruita a partire dalla versione senza.

QUERY 14 (versione 1)

**select distinct artist.name,count(release.name)**

**from release**

**join artist\_credit**

**on release.artist\_credit=artist\_credit.id**

**join artist\_credit\_name**

**on artist\_credit.id=artist\_credit\_name.artist\_credit**

**join artist**

**on artist\_credit\_name.artist=artist.id**

**join medium**

**on release.id=medium.release**

**join medium\_format**

**on medium.format=medium\_format.id**

**where medium\_format.name='CD'**

**and artist.name not in**

**(**

**select distinct artist.name**

**from medium join track**

**on track.medium=medium.id**

**join medium\_format**

**on medium.format=medium\_format.id**

**join release**

**on medium.release=release.id**

**join artist\_credit**

**on release.artist\_credit=artist\_credit.id**

**join artist\_credit\_name**

**on artist\_credit.id=artist\_credit\_name.artist\_credit**

**join artist**

**on artist\_credit\_name.artist=artist.id**

**where medium\_format.name='CD'**

**and medium.track\_count<=all**

**(**

**select avg(medium.track\_count)**

**from medium**

**join track**

**on track.medium=medium.id**

**join medium\_format**

**on medium.format=medium\_format.id**

**join release**

**on medium.release=release.id**

**where medium\_format.name='CD'**

**)**

**)**

**group by artist.name**

**order by count(release.name) desc**

**1)** Nel risultato vengono richiesti il nome dell’artista ( attributo name della tabella artist) e il numero delle release, ordinato per numero delle release decrescente.

**2**) Il primo join è tra la chiave primaria della tabella medium e la chiave esterna medium ella tabella track, il successivo è tra la chaive esterna format della tabella medium e la chiave primaria della tabella medium\_format. In seguito c’è il join tra la chiave esterna release della tabella medium e la chiave primaria della tabella release. C’è poi il join tra la chiave primaria di artist\_credit e la chiave esterna artist\_credit della tabella release. L’altro join è tra le chiavi primarie di artist\_credit e artist\_credit\_name.L’ultimo è il join tra la chiave primaria id della tabella artist con la chiave esterna artist della tabella artist\_credit\_name.

Per quanto riguarda il calcolo della media sono stati fatti gli stessi join a parte quelli con le tabelle artist,artist\_credit,artist\_credit\_name.

**3)** In questa query sono presenti 2 sotto problemi, il più interno serve per calcolare la media delle tracce, il secondo sotto problema più interno serve invece per trovare tutti gli artisti che hanno release con lunghezza minore della media.

**4)** Si è usato il distinct per i duplicati

**5)** Sono richieste due versioni della query.

**6)**

**select distinct artist.name,count(release.name)**

**from release**

**join artist\_credit**

**on release.artist\_credit=artist\_credit.id**

**join artist\_credit\_name**

**on artist\_credit.id=artist\_credit\_name.artist\_credit**

**join artist**

**on artist\_credit\_name.artist=artist.id**

**join medium**

**on release.id=medium.release**

**join medium\_format**

**on medium.format=medium\_format.id**

**where medium\_format.name='CD'**

**and artist.name not in**

**(**

**select distinct artist.name**

**from medium join track**

**on track.medium=medium.id**

**join medium\_format**

**on medium.format=medium\_format.id**

**join release**

**on medium.release=release.id**

**join artist\_credit**

**on release.artist\_credit=artist\_credit.id**

**join artist\_credit\_name**

**on artist\_credit.id=artist\_credit\_name.artist\_credit**

**join artist**

**on artist\_credit\_name.artist=artist.id**

**where medium\_format.name='CD'**

**and medium.track\_count <= all**

**(**

**select avg(medium.track\_count)**

**from medium**

**join track**

**on track.medium=medium.id**

**join medium\_format**

**on medium.format=medium\_format.id**

**join release**

**on medium.release=release.id**

**where medium\_format.name='CD'**

**)**

**)**

**group by artist.name**

**intersect**

**select distinct artist.name,count(release.name)**

**from release**

**join artist\_credit**

**on release.artist\_credit=artist\_credit.id**

**join artist\_credit\_name**

**on artist\_credit.id=artist\_credit\_name.artist\_credit**

**join artist**

**on artist\_credit\_name.artist=artist.id**

**join medium**

**on release.id=medium.release**

**join medium\_format**

**on medium.format=medium\_format.id**

**where medium\_format.name='CD'**

**and artist.name in**

**(**

**select distinct artist.name**

**from medium join track**

**on track.medium=medium.id**

**join medium\_format**

**on medium.format=medium\_format.id**

**join release**

**on medium.release=release.id**

**join artist\_credit**

**on release.artist\_credit=artist\_credit.id**

**join artist\_credit\_name**

**on artist\_credit.id=artist\_credit\_name.artist\_credit**

**join artist**

**on artist\_credit\_name.artist=artist.id**

**where medium\_format.name='CD'**

**and medium.track\_count <= all**

**(**

**select avg(medium.track\_count)**

**from medium**

**join track**

**on track.medium=medium.id**

**join medium\_format**

**on medium.format=medium\_format.id**

**join release**

**on medium.release=release.id**

**where medium\_format.name='CD'**

**)**

**)**

**group by artist.name**

Questa query cerca l’intersezone tra l’insieme degli artisti che hanno tutte le registrazioni maggiori della media e gli altri. L’intersezione è nulla quindi la query è giusta.

QUERY 14 (versione 2)

**create view Media\_t as**

**select avg(medium.track\_count)**

**from medium**

**join track**

**on track.medium=medium.id**

**join medium\_format**

**on medium.format=medium\_format.id**

**join release**

**on medium.release=release.id**

**where medium\_format.name='CD'**

**create view ArtMinore as**

**select distinct artist.name**

**from medium join track**

**on track.medium=medium.id**

**join medium\_format**

**on medium.format=medium\_format.id**

**join release**

**on medium.release=release.id**

**join artist\_credit**

**on release.artist\_credit=artist\_credit.id**

**join artist\_credit\_name**

**on artist\_credit.id=artist\_credit\_name.artist\_credit**

**join artist**

**on artist\_credit\_name.artist=artist.id**

**where medium\_format.name='CD'**

**and medium.track\_count<=all**

**(**

**select \* from Media\_t**

**)**

**group by artist.name**

**create view AllArt as**

**select distinct artist.name, count(release.name)**

**from release**

**join artist\_credit**

**on release.artist\_credit=artist\_credit.id**

**join artist\_credit\_name**

**on artist\_credit.id=artist\_credit\_name.artist\_credit**

**join artist**

**on artist\_credit\_name.artist=artist.id**

**join medium**

**on release.id=medium.release**

**join medium\_format**

**on medium.format=medium\_format.id**

**where medium\_format.name='CD'**

**group by artist.name**

**select \* from AllArt**

**where AllArt.name not in**

**(**

**select \* from ArtMinore**

**)**

**order by count desc**

**1)** Le richieste sono le stesse della versione 1.

**2)** I join sono gli stessi fatti per la versione 1.

**3)** I sotto problemi sono gli stessi della prima query, sono state create 3 viste, una per calcolare la media(Media\_t), una per trovare tutti gli artist con il numero delle release(ArtAll) e un’altra per trovare tutti gli artisti con registrazioni sotto la media(ArtMinore.

**4)** Il discorso sui duplicati e sui valori nulli vale lo stessso fatto per la versione 1.

**5)** E’ già stata fatta una prima versione della query.

**6)** La query è stata fatta con le viste sulla base della versione 1, quindi la verifica fatta per la versione 1 vale anche per questa .

QUERY 15 (versione 1)

**select**

**query.name,query.begin\_date\_day,query.begin\_date\_month,query.begin\_date\_year,query.end\_date\_day,**

**query.end\_date\_month,query.end\_date\_year**

**from**

**(**

**select artist.name as name,artist.begin\_date\_day,artist.begin\_date\_month,artist.begin\_date\_year,artist.end\_date\_day,artist.end\_date\_month,artist.end\_date\_year**

**from artist**

**where artist.end\_date\_day >**

**(**

**select artist.end\_date\_day**

**from artist**

**where artist.name = 'Louis Armstrong'**

**)**

**and artist.end\_date\_month >**

**(**

**select artist.end\_date\_month**

**from artist**

**where artist.name = 'Louis Armstrong'**

**)**

**and artist.end\_date\_year >**

**(**

**select artist.end\_date\_year**

**from artist**

**where artist.name = 'Louis Armstrong'**

**)**

**)query**

**where query.end\_date\_day =**

**(**

**select min(query.end\_date\_day)**

**from**

**(**

**select artist.name as name,artist.end\_date\_day,artist.end\_date\_month,artist.end\_date\_year**

**from artist**

**where artist.end\_date\_day >**

**(**

**select artist.end\_date\_day**

**from artist**

**where artist.name = 'Louis Armstrong'**

**)**

**and artist.end\_date\_month >**

**(**

**select artist.end\_date\_month**

**from artist**

**where artist.name = 'Louis Armstrong'**

**)**

**and artist.end\_date\_year >**

**(**

**select artist.end\_date\_year**

**from artist**

**where artist.name = 'Louis Armstrong'**

**)**

**)query**

**)**

**and query.end\_date\_month =**

**(**

**select min(query.end\_date\_month)**

**from**

**(**

**select artist.name as name,artist.end\_date\_day,artist.end\_date\_month,artist.end\_date\_year**

**from artist**

**where artist.end\_date\_day >**

**(**

**select artist.end\_date\_day**

**from artist**

**where artist.name = 'Louis Armstrong'**

**)**

**and artist.end\_date\_month >**

**(**

**select artist.end\_date\_month**

**from artist**

**where artist.name = 'Louis Armstrong'**

**)**

**and artist.end\_date\_year >**

**(**

**select artist.end\_date\_year**

**from artist**

**where artist.name = 'Louis Armstrong'**

**)**

**)query**

**)**

**and query.end\_date\_year =**

**(**

**select min(query.end\_date\_year)**

**from**

**(**

**select artist.name as name,artist.end\_date\_day,artist.end\_date\_month,artist.end\_date\_year**

**from artist**

**where artist.end\_date\_day >**

**(**

**select artist.end\_date\_day**

**from artist**

**where artist.name = 'Louis Armstrong'**

**)**

**and artist.end\_date\_month >**

**(**

**select artist.end\_date\_month**

**from artist**

**where artist.name = 'Louis Armstrong'**

**)**

**and artist.end\_date\_year >**

**(**

**select artist.end\_date\_year**

**from artist**

**where artist.name = 'Louis Armstrong'**

**)**

**)query**

**)**

La query ricava tutte le informazione del caso solo dall’entità artist.

la query prende dall’entità artist il nome e gli attributi riguardanti la data di morte dell’artista,non vi sono join da trattare mentre possibili valori nulli riguardano gli attributi della data di morte in quanto gli artisti potrebbero essere ancora vivi.

Il primo sottoproblema permette di trovare tutti gli artisti morti dopo Louis Armstrong in cui vi sono 3 ulteriori sottoproblemi in cui si ricavano giorno, mese e anno di morte d’ogni artista. Altre tre sottoquery servono per trovare, tra gli artisti ricavati prima, quello morto dopo Louis Armstrong (rispettivamente ricavano giorno, mese e anno minimo tra gli artisti ricavati prima).

Tutti i valori nulli vengono automaticamente esclusi dalla query quando si va a verificare se l’artista è morto o meno e , nel caso lo sia, se prima o dopo Louis Armstrong.

é presente una seconda versione della query in cui vengono utilizzate le viste.

**select query.name,query.end\_date\_day,query.end\_date\_month,query.end\_date\_year**

**from**

**(**

**select artist.name as name,artist.end\_date\_day,artist.end\_date\_month,artist.end\_date\_year**

**from artist**

**where artist.end\_date\_day >**

**(**

**select artist.end\_date\_day**

**from artist**

**where artist.name = 'Louis Armstrong'**

**)**

**and artist.end\_date\_month >**

**(**

**select artist.end\_date\_month**

**from artist**

**where artist.name = 'Louis Armstrong'**

**)**

**and artist.end\_date\_year >**

**(**

**select artist.end\_date\_year**

**from artist**

**where artist.name = 'Louis Armstrong'**

**)**

**)query**

**order by query.end\_date\_year,query.end\_date\_month,query.end\_date\_day**

La query di verifica restituisce tutti gli artisti morti dopo Louis Armstrong e dato che sono ordinati in ordine crescente, il primo risultato sarà il primo artista morto dopo di lui e coincide con il risultato della query richiesta. SI conclude così che la query richiesta è corretta.

QUERY 15 (versione 2)

**create view DayDeath as**

**select artist.end\_date\_day**

**from artist**

**where artist.name = 'Louis Armstrong'**

**create view MonthDeath as**

**select artist.end\_date\_month**

**from artist**

**where artist.name = 'Louis Armstrong'**

**create view YearDeath as**

**select artist.end\_date\_year**

**from artist**

**where artist.name = 'Louis Armstrong'**

**create view DeathAfterArm as**

**select artist.name as name,artist.end\_date\_day,artist.end\_date\_month,artist.end\_date\_year**

**from artist**

**where artist.end\_date\_day >**

**(**

**select \***

**from DayDeath**

**)**

**and artist.end\_date\_month >**

**(**

**select \***

**from MonthDeath**

**)**

**and artist.end\_date\_year >**

**(**

**select \***

**from YearDeath**

**)**

**select query.name,query.begin\_date\_day,query.begin\_date\_month,query.begin\_date\_year,query.end\_date\_day,query.end\_date\_month,query.end\_date\_year**

**from**

**(**

**select artist.name as name,artist.begin\_date\_day,artist.begin\_date\_month,artist.begin\_date\_year,artist.end\_date\_day,artist.end\_date\_month,artist.end\_date\_year**

**from artist**

**where artist.end\_date\_day >**

**(**

**select \***

**from DayDeath**

**)**

**and artist.end\_date\_month >**

**(**

**select \***

**from MonthDeath**

**)**

**and artist.end\_date\_year >**

**(**

**select \***

**from YearDeath**

**)**

**)query**

**where query.end\_date\_day =**

**(**

**select min(DeathAfterArm.end\_date\_day)**

**from DeathAfterArm**

**)**

**and query.end\_date\_month =**

**(**

**select min(DeathAfterArm.end\_date\_month)**

**from DeathAfterArm**

**)**

**and query.end\_date\_year =**

**(**

**select min(DeathAfterArm.end\_date\_year)**

**from DeathAfterArm**

**)**

1. La query ricava tutte le informazione del caso solo dall’entità artist.
2. la query prende dall’entità artist il nome e gli attributi riguardanti la data di morte dell’artista,non vi sono join da trattare mentre possibili valori nulli riguardano gli attributi della data di morte in quanto gli artisti potrebbero essere ancora vivi.
3. questo punto è uguale al precedente punto 3 dell’altra versione della query.
4. Tutti i valori nulli vengono automaticamente esclusi dalla query quando si va a verificare se l’artista è morto o meno e , nel caso lo sia, se prima o dopo Louis Armstrong.
5. é presente una seconda versione della query in cui vengono utilizzate le viste.
6. La modalità di controllo è analoga a quella della prima versione della query