# Database ATP Regular Season 2015


Francesco Monti

# INDICE

## PARTE SQL



## PARTE MONGODB



## JAVA

# MYSQL

## DESCRIZIONE

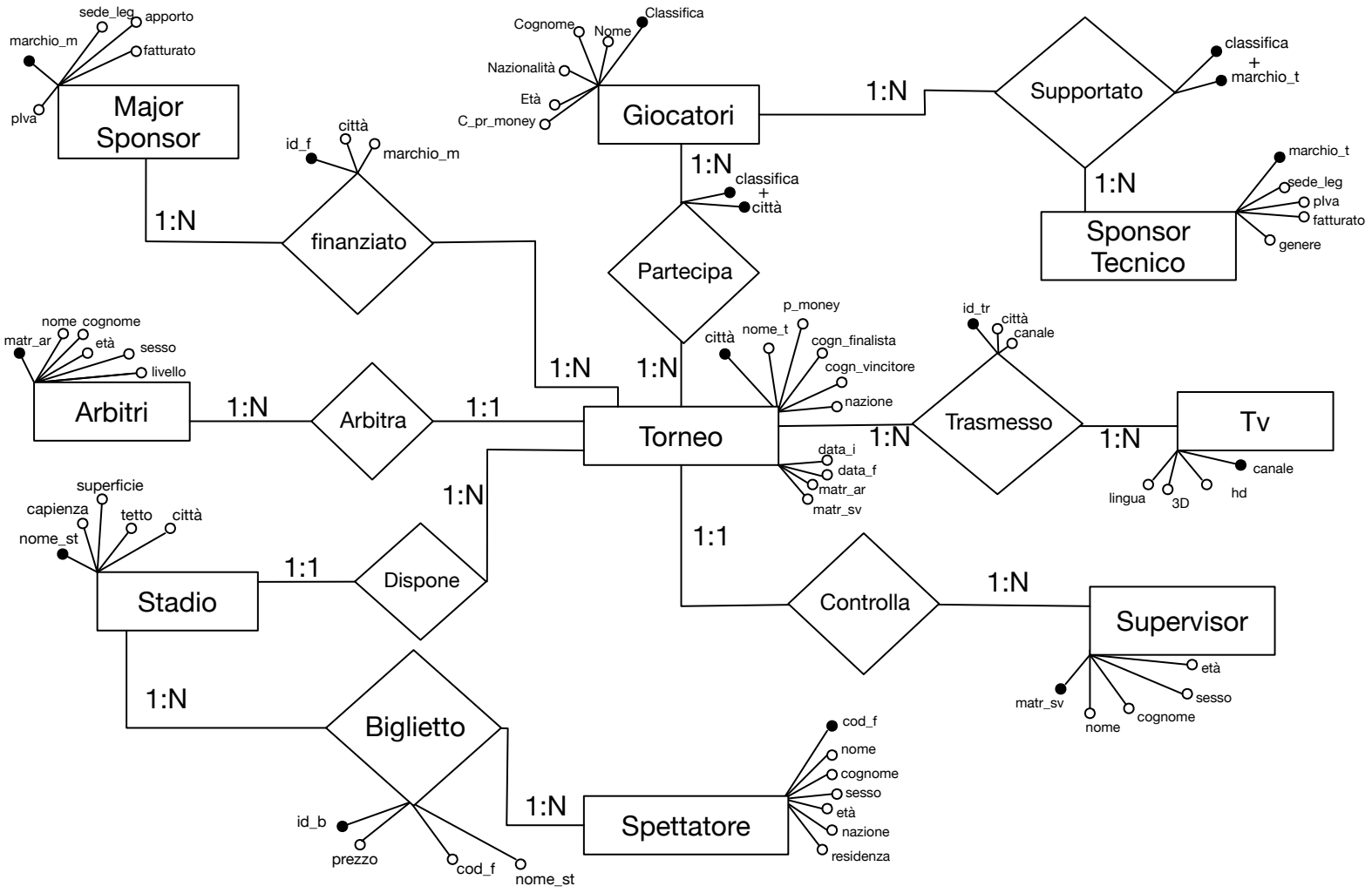
Si vuole rappresentare la passata stagione del circuito ATP , in un DB che contenga i dati necessari per analizzarla.
Il DB deve rappresentare opportunamente, i tornei svolti durante l' annata ed i loro partecipanti, con particolare interesse verso i finalisti.

Va accuratamente gestita la suddivisione degli sponsor che, andremo a collocare in due macro categorie:
**Sponsor Tecnici** : associati spesso al singolo giocatore. Descrivono collaborazioni riguardanti abbigliamento o materiale di ogni genere necessario alla pratica sportiva considerata.
**Sponsor Major** : sono gli sponsor che finanziano i tornei in giro per il mondo e ne garantiscono svolgimento organizzazione ed appetibilità, grazie al loro apporto.

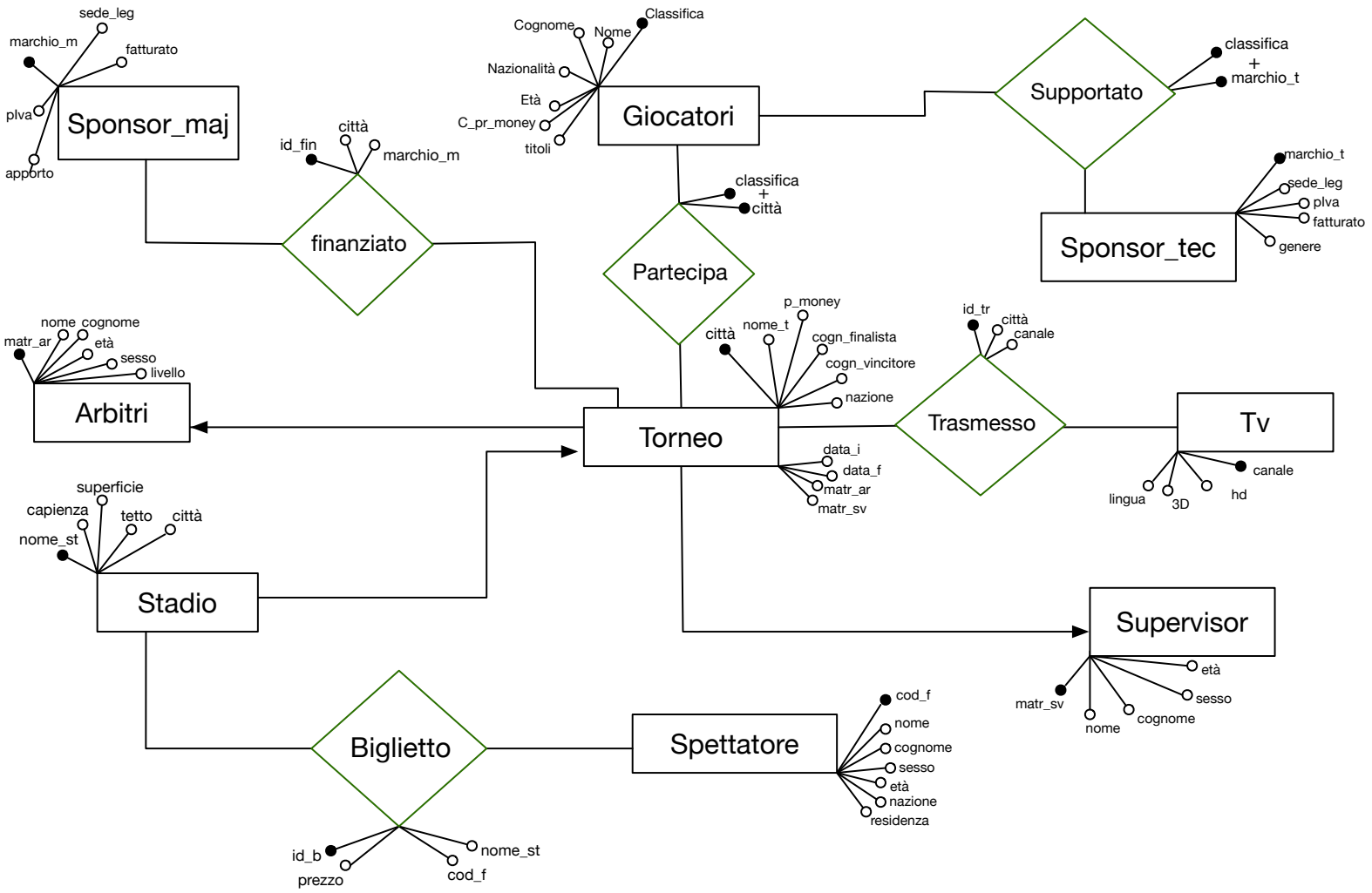
Inoltre, si vuole poter' estrapolare, dati sugli arbitraggi delle finali e sugli arbitri stessi. Ai fini di alcune ricerche, e dell assegnamento dei diritti tv per la stagione 2016, interessano i dati sui palinsesti delle tv mondiali che trasmettono i tornei del circuito.

Non va tralasciato l aspetto economico dei tornei relativo alle entrate, provenienti dai biglietti acquistati , e le conseguenti analisi di mercato che implicano la conoscenza dei dati minimi per ogni singolo spettatore.

# Schema E/R Logico



# E/R Normalizzato/Fisico



# CREAZIONE TABELLE

Visto che avremo diverse chiavi secondarie, ci interessa fare in modo di inserire parecchi dati relativi ad entità spettatore e biglietti, insieme , opteremo per l'utilizzo dello Storage Engine InnoDB. Dato che i nostri dati servono per  analisi di mercato per la stagione 2016 e per considerazioni sulla stagione appena conclusa. Inoltre per “sicurezza”  decidiamo di non permettere la rimozione di una foreign key senza la precedente eliminazione dei dati ad essa collegati.

## Specifiche e considerazioni prima della creazione

- Le partite iva vengono scelte e rappresentate tramite codice numerico ad 11 cifre
- Gli apporti degli sponsor Major per singolo torneo non superano mai il milione
- Il limite per i singoli fatturati aziendali degli sponsor ed i career price money, è quello del tipo di dato Integer.
- I codici fiscali vengono considerati universali, rappresentati come,  stringhe alfanumeriche di 12 caratteri.
- Gli apporti degli sponsor major sono fissi ed uguali per ogni torneo
- Gli sponsor Major BNL  e  Fly-Emirates finanziano ogni torneo
- Le nazionalità sono espresse in modo abbreviato con i consueti 3 caratteri (ITA,FRA,…)
- Possiamo usare come chiave primaria della tabella giocatore la classifica in quanto a fine stagione non ci saranno cambiamenti
- Vengono considerati appartenenti al circuito ATP i primi 250 giocatori del mondo
- Non esiste giocatore con più di 90 titoli vinti all’ attivo
- Non esiste giocatore con un career price money superiore ai 90 milioni
- Il price money di un torneo nn supera mai i 5 milioni
- I tornei assumono come nome il nome della città ospitante concatenato con uno degli identificativi del circuito Atp (-tennis , -tour, -master, -ace, -world , -cup )
- Non ci interessa il campo data nei biglietti in quanto,  le analisi dell ATP sono “macroanalisi” che non si occuperanno delle singolo giornate nei singoli tornei
- I prezzi dei biglietti sono dettati dall’ Atp e raggiungono un prezzo massimo pari a 250 euro per biglietto in una sessione

Di seguito la creazione delle nostre entità riportate nel corretto ordine d' inserimento nel DB.

## 1) ENTITA' SPONSOR TECNICO record : 150

Quest' entità rappresenta gli sponsor tecnici, ovvero , quegli sponsor che oltre al supporto economico , forniscono materiale per la pratica sportiva.
vengono rappresentati i campi che descrivono : nome del marchio(Pk)
( marchio_t) , la sede legale , la partita Iva , il genere ed il fatturato annuo aziendale.

```
create table sponsor_tec(
     marchio_t varchar(20) not null primary key,
     sede_leg  varchar(20) not null,
     p_iva char(11) not null,
     genere varchar(15) not null,
     fatturato integer not null
);
```

UN INSERT TIPO :

```
insert into sponsor_tec values('Nike','Milano','104-48-7066','abbigliamento',107611761);
```

## 2) ENTITA' MAJOR SPONSOR record : 100

Quest' entità rappresenta i Major sponsor, ovvero , quegli sponsor che con il loro apporto economico aiutano l organizzazione e la buona riuscita dei tornei. vengono rappresentati i campi che descrivono : nome del marchio (Pk) ( marchio_m) ,
la sede legale , la partita Iva , l' apporto ed il fatturato annuo aziendale.

```
create table sponsor_maj(
     marchio_m varchar(20) not null primary key,
     sede_leg varchar(20) not null,
     p_iva varchar(11) not null,
     apporto mediumint not null,
     fatturato integer not null
);
```

UN INSERT TIPO :

```
insert into sponsor_maj values ('BNL','Paris', '299-62-6211', 57896, 718808916);
```

### 3) ENTITA' TV  record : 50

Quest' entità rappresenta le tv che trasmettono le giornate del torneo. i campi rappresentati sono: canale(Pk)  , hd , 3d , lingua.

```
create table tv(
     canale varchar(20) not null primary key,
     hd boolean not null,
     3d boolean not null,
     lingua varchar(15) not null
);
```

UN INSERT TIPO :

```
insert into tv values ('sky', true, true, 'Polish');
```

### 4) ENTITA'  ARBITRI  record : 250

Quest' entità rappresenta gli arbitri che sono stati designati per la stagione 2015 Atp. I campi che li descriveranno sono: la matricola_ar(Pk)  rappresentata da un numero da 1 a 250 concatenato con il char 'A',  nome , cognome , livello ( A,B,C,D,E ) , età e sesso.

```
create table arbitri(
     matricola_ar varchar(4) not null primary key,
     nome varchar(15) not null,
     cognome varchar(15) not null,
     livello char(1) not null,
     eta tinyint(2) not null,
     sesso char(1) not null
);
```

UN INSERT TIPO :

```
 insert into arbitri values ('249A' , 'Mohammed' , 'Lhaiani' , 'A' , 56 , 'M');
```

## 5) ENTITA’ SUPERVISOR record : 250

Quest’ entità rappresenta i supervisor cioè , quelle persone addette alla supervisione dell andamento dei match durante tutto il torneo. Rappresentiamo l entità con i campi : matricola_sv(Pk) rappresentata da un numero da 1 a 250 concatenato con il char ‘S’, nome , cognome , età e sesso.

```
create table supervisor(
     matricola_sv varchar(4) not null primary key,
     nome varchar(15) not null,
     cognome varchar(15) not null,
     eta tinyint(2) not null,
     sesso char(1) not null
);
```

UN INSERT TIPO :

```
insert into supervisor values (‘1S’, ‘Walter’, ‘Collins’, 47, ‘M’);
```

## 6) ENTITA’ GIOCATORE record : 250

Quest’ entità rappresenta i primi 250 giocatori del mondo. I campi che li rappresentano sono : classifica(Pk), nome , cognome , eta , nazione , titoli e car_p_money (career price money).

```
create table giocatore(
     classifica tinyint unsigned not null primary key,
     nome varchar(15) not null,
     cognome varchar(15) not null,
     eta tinyint(2) unsigned not null,
     nazione char(3) not null,
     titoli tinyint(2) unsigned not null,
     car_p_money integer(8) not null /* max 90 milioni */
);
```

UN INSERT TIPO :

```
insert into giocatore values (3,'Roger' , 'Federer' , ’35', ‘SUI’,90, 89000000);
```

## 7) ENTITA’ SPETTATORE record : 100.000

Quest’ entità rappresenta un campione vasto di spettatori registrati in giro per i tornei del circuito Atp. Essi sono rappresentati dai seguenti campi: cod_f(Pk) ( codice fiscale) , nome , cognome, età, nazione, sesso e residenza.

```
create table spettatore(
     cod_f char(12) primary key not null,
     nome varchar(15) not null,
     cognome varchar(15) not null,
     eta tinyint(2) not null,
     nazione varchar(3) not null,
     sesso char(1) not null,
     residenza varchar(20) not null
);
```

UN INSERT TIPO :

```
insert into spettatore values ( 'mntfnc12892abr','Francesco','Monti',21,'ITA','M','Broccostella');
```

## 8) ENTITA’ TORNEO record : 100

Quest’ entità rappresenta il singolo torno. Descritto con: città(Pk), nome_t , p_money, cogn_finalista ( cognome finalista ), cogn_vincitore, nazione, data_i e data_f (inizio e fine), matricola_ar e matricola_sv ( Fk).

```
create table torneo(
     citta varchar(20) primary key not null,
     nome_t varchar(30) not null,
     p_money integer(7) not null,
     cogn_finalista varchar(15) not null,
     cogn_vincitore varchar(15) not null,
     nazione char(3) not null,
     data_i date not null,
     data_f date not null,
     matricola_ar varchar(4) not null ,
     foreign key (matricola_ar) references
     arbitri(matricola_ar) on update cascade on delete no action,
     matricola_sv varchar(4) not null ,
     foreign key (matricola_sv) references
     supervisor(matricola_sv)on update cascade on delete no action
);
```

UN INSERT TIPO :

```
insert into torneo  values (‘Roma’, ‘Roma-world’, 4006533, ‘Cilic’, ‘Kudla’,’ITA’,
                                                ’2015-1-1’, ‘2015-1-7’, ‘72A’, ‘191S’;
```

## 9) ENTITA’ STADIO record : 358

Quest’ entità rappresenta e descrive i singoli stadi per ogni torneo. Essi sono descritti con: nome_st(Pk), capienza, superficie, tetto, città(Fk).

```
create table stadio(
     nome_st varchar(20) primary key not null,
     capienza smallint not null,
     superficie varchar(7) not null,
     tetto boolean not null,
     citta varchar(20) not null ,
     foreign key (citta) references torneo(citta) on update
     cascade on delete no action
);
```

UN INSERT TIPO :

```
insert into stadio values ( ‘Roma-One’, ‘1191’, ‘terra’, 0,  ‘Roma’ );
```

## 10) ENTITA’ SUPPRTATO record : 500

Quest’ entità rappresenta il collegamento fra giocatori ed i loro sponsor tecnici. Essi sono descritti con: marchio_t e classifica che insieme formano la chiave primaria per quest’ entità.

```
create table supportato(
     marchio_t varchar(20) not null,
     foreign key (marchio_t) references sponsor_tec(marchio_t) on
     update cascade on delete no action,
     classifica tinyint unsigned not null,
     foreign key (classifica) references giocatore(classifica) on
     update cascade on delete no action,
     primary key ( marchio_t, classifica)
);
```

UN INSERT TIPO :

```
insert into supportato values ( ‘Nike', '2');
```

## 11) ENTITA’ FINANZIATO record : 4999

Quest’ entità rappresenta i collegamenti che ci sono tra i tornei ed i major-sponsor, per analizzare quale torneo è sponsorizzato da quale sponsor. L’ entità è cosi formata: id_fin (Pk)( numero intero concatenato con char ‘F’), e le chiavi secondarie marchio_m e città.

```
create table finanziato(
     id_fin varchar(5) primary key not null,
     marchio_m varchar(20) not null,
     foreign key (marchio_m) references sponsor_maj(marchio_m)
     on update cascade on delete no action,
     citta varchar(20) not null,
     foreign key (citta) references torneo(citta) on update
     cascade on delete no action
);
```

UN INSERT TIPO :

```
insert into finanziato values ( ‘1F’, ‘BNL', 'Roma');
```

## 12) ENTITA’ TRASMESSO record : 4999

Quest’ entità rappresenta i collegamenti che ci sono tra i tornei e le trasmissioni effettuate dai diversi canali. L’ entità è cosi formata: id_tr (Pk)( numero intero concatenato con char ‘T’), e le chiavi secondarie città e canale.

```
create table trasmesso(
     id_tr varchar(5) primary key not null,
     canale varchar(20) not null,
     foreign key (canale) references tv(canale) on update cascade
     on delete no action,
     citta varchar(20) not null,
     foreign key (citta) references torneo(citta) on update
     cascade on delete no action
);
```

UN INSERT TIPO :

```
insert into trasmesso values ( ‘1T', ‘sky', 'Roma');
```

## 13) ENTITA’ PARTECIPA record : 14236

Quest’ entità rappresenta le partecipazioni dei singoli giocatori ai tornei presenti nel circuito nella stagione 2015. Componiamo l entità: classifica e città (Pk), unite a formare la chiave primaria utile per la ‘navigazione’ nel DB.

```
create table partecipa(
     classifica tinyint unsigned not null,
     foreign key (classifica) references giocatore(classifica)
     on update cascade on delete no action,
     citta varchar(20) not null,
     foreign key (citta) references torneo(citta) on update
     cascade on delete no action,
     primary key( classifica , citta)
);
```

UN INSERT TIPO :

```
insert into partecipa values ( ‘1', 'Roma');
```

## 14) ENTITA’ BIGLIETTO record : 250.000

Quest’ entità rappresenta i biglietti relativi agli spettatori. Per ogni biglietto teniamo conto di: id_b (massimo 6 cifre concatenate con char ‘B’ (Pk), prezzo, cod_f e nome_st (Fk).

```
create table biglietto(
     id_b varchar(7) primary  key not null ,
     prezzo tinyint unsigned not null,
     cod_f char(12) not null,
     foreign key (cod_f) references spettatore(cod_f) on    update
     cascade on delete no action,
     nome_st varchar(20) not null,
     foreign key (nome_st) references stadio(nome_st) on update
     cascade on delete no action
);
```

UN INSERT TIPO :

```
insert into biglietto values ( ‘1B', 60, ‘Xg0RqLQFZAnG', ‘Bastad-One');
```

## Sponsor_maj

| marchio_m | sede_leg | p_iva | apporto | fatturato |
|---|---|---|---|---|
| AB | Atene | 129-18-5399 | 953933 | 789094075 |
| Algida | Guangzhou | 538-87-6550 | 740693 | 56072208 |
| Alitalia | Dublino | 695-60-3555 | 999543 | 203784603 |
| Allianz | Krakowia | 822-10-3182 | 577197 | 733837243 |
| Asics | Milano | 807-64-4514 | 149294 | 742137331 |

## Sponsor_tec

| marchio_t | sede_leg | p_iva | genere | fatturato |
|---|---|---|---|---|
| Adidas | Berlino | 562-44-3569 | abbigliamento | 96305682 |
| Ailane | Berlino | 403-14-2813 | accessori | 6399742 |
| Aimbo | Basilea | 261-04-5916 | integratori | 6284315 |
| Ainyx | NewYork | 763-98-9879 | abbigliamento | 1089202 |
| Aivee | Roma | 521-25-0355 | accessori | 9487692 |

## TV

| canale | hd | 3d | lingua |
|---|---|---|---|
| acetv | 0 | 1 | Polish |
| acetv2 | 0 | 1 | Italian |
| bigtv1 | 1 | 0 | Frances |
| bigtv2 | 1 | 0 | Polish |
| bigtv3 | 1 | 1 | Polish |

## Arbitri

| matricola_ar | nome | cognome | livello | eta | sesso |
|---|---|---|---|---|---|
| 100A | Karen | Adams | A | 44 | F |
| 101A | Charles | Alvarez | D | 47 | M |
| 102A | Karen | Johnson | C | 42 | F |
| 103A | Lillian | Diaz | B | 42 | F |
| 104A | Louis | Warren | B | 49 | M |

## Supervisor

| matricola_sv | nome | cognome | eta | sesso |
|---|---|---|---|---|
| 100S | Brandon | Thompson | 56 | M |
| 101S | Harry | Thomas | 62 | M |
| 102S | Mildred | Simpson | 63 | F |
| 103S | Kelly | Howell | 57 | F |
| 104S | Theresa | Ryan | 52 | F |

## Giocatore

| classifica | nome | cognome | eta | nazione | titoli | car_p_money |
|---|---|---|---|---|---|---|
| 1 | Novak | Djokovic | 28 | SRB | 54 | 75000000 |
| 2 | Andy | Murray | 28 | GBR | 23 | 56070000 |
| 3 | Roger | Federer | 34 | SUI | 90 | 90000000 |
| 4 | Stan | Wawrinka | 31 | SUI | 22 | 22000000 |
| 5 | Rafael | Nadal | 29 | ESP | 70 | 84000000 |

## Spettatore

| cod_f | nome | cognome | eta | nazione | sesso | residenza |
|---|---|---|---|---|---|---|
| 002Pz4wNKvMv | Julie | Frazier | 52 | MEX | F | Oslo |
| 004gMgQjhg1C | Debra | Edwards | 38 | ESP | F | Spencer |
| 006VHHlgD5Wh | Pamela | Tucker | 54 | FRA | F | Katowice |
| 008EM7hGWILy | Martin | Mcdonald | 52 | FRA | M | Hohhot |
| 00Ashd82wDJG | Ruby | Welch | 73 | USA | F | Doha |

## Torneo

| citta | nome_t | p_money | cogn_finalista | cogn_vincitore | nazione | data_i | data_f | matricola_ar | matricola_sv |
|---|---|---|---|---|---|---|---|---|---|
| Anning | Anning-ace | 3210187 | Dolgopolov | Federer | GER | 2015-08-28 | 2015-09-03 | 77A | 135S |
| Antwerp | Antwerp-tour | 1908722 | Bolelli | Federer | ENG | 2015-08-01 | 2015-08-07 | 62A | 122S |
| Atene | Atene-master | 3383118 | Sousa | Raonic | GRE | 2015-02-01 | 2015-02-07 | 157A | 247S |
| Atlanta | Atlanta-world | 871749 | Dolgopolov | Nadal | USA | 2015-07-16 | 2015-07-22 | 150A | 150S |
| Baotou | Baotou-tennis | 4891190 | Lorenzi | Djokovic | CHN | 2015-02-13 | 2015-02-19 | 33A | 102S |

## Stadio

| nome_st | capienza | superficie | tetto | citta |
|---|---|---|---|---|
| Anning-One | 4974 | erba | 0 | Anning |
| Anning-Three | 5435 | erba | 1 | Anning |
| Anning-Two | 3227 | erba | 1 | Anning |
| Antwerp-One | 6001 | erba | 1 | Antwerp |
| Antwerp-Three | 8717 | erba | 1 | Antwerp |

## Supportato

| marchio_t | classifica |
|---|---|
| Adidas | 1 |
| Head | 1 |
| Roodel | 1 |
| Artengo | 2 |
| Nike | 2 |

## Finanziato

| id_fin | marchio_m | citta |
|---|---|---|
| 1000F | Ferrari | Napoli |
| 1001F | Costa | Marburg |
| 1002F | Wind | Nizza |
| 1003F | Squib | Halle |
| 1004F | Garnier | Umago |

## Trasmesso

| id_tr | canale | citta |
|---|---|---|
| 1000T | bigtv1 | New Even |
| 1001T | cnn | Berlino |
| 1002T | canal3 | Oslo |
| 1003T | tv4 | Miami |
| 1004T | srf2 | Basilea |

## Partecipa

| classifica | citta |
|---|---|
| 1 | Anning |
| 2 | Anning |
| 3 | Anning |
| 4 | Anning |
| 5 | Anning |

## Biglietto

| id_b | prezzo | cod_f | nome_st |
|---|---|---|---|
| 100000B | 90 | mntfnc12892a | Torino-One |
| 100001B | 90 | ZrFQtpYn0UJo | Winnipeng-Three |
| 100002B | 30 | sE7Xvhd8WP0a | Maskor-Three |
| 100003B | 210 | ExpAVHfiK4RB | Noumea-Three |
| 100004B | 60 | e9Ej7wVLjdQa | NewEven-One |

# QUERY

Le seguenti interrogazioni, rappresentano probabili estrazioni di dati utili nella realtà a fini statistici o di marketing, cercando di coprire più richieste possibili, mantenendo un buon metodo di scrittura e non dimenticando le prestazioni nel caso di query “pesanti”.

## 1) SELEZIONARE TUTTI I VINCITORI DELLA STAGIONE 2015, ELENCANDO CLASSIFICA , COGNOME , NOME E NAZIONE.

```
SELECT giocatore.classifica, torneo.cogn_vincitore,giocatore.nome,
       giocatore.nazione
FROM torneo JOIN partecipa ON torneo.citta = partecipa.citta
JOIN giocatore ON partecipa.classifica = giocatore.classifica
WHERE torneo.cogn_vincitore = giocatore.cognome
GROUP BY giocatore.cognome
ORDER BY giocatore.classifica;
```

| classifica | cogn_vincitore | nome | nazione |
|---|---|---|---|
| 1 | Djokovic | Novak | SRB |
| 2 | Murray | Andy | GBR |
| 3 | Federer | Roger | SUI |
| 4 | Wawrinka | Stan | SUI |
| 5 | Nadal | Rafael | ESP |
| 6 | Nishikori | Kei | JPN |
| 7 | Berdych | Tomas | CZE |
| 8 | Ferrer | David | ESP |
| 12 | Raonic | Milos | CAN |
| 14 | Thiem | Dominic | AUT |
| 26 | Dimitrov | Grigor | BUL |
| 48 | Coric | Borna | CRO |
| 54 | Zverev | Alexander | GER |
| 57 | Kudla | Denis | USA |
| 58 | Vesely | Jiri | CZE |

15 rows in set (0,02 sec)

## 2) SELEZIONARE NOME ,COGNOME ,CLASSIFICA DEL NUMERO 1 , SPECIFICANDO QUANTI SONO I TORNEI DA LUI VINTI IN STAGIONE

```
SELECT count(*) "Vittorie 2015", giocatore.classifica,
giocatore.nome, giocatore.cognome
FROM torneo JOIN partecipa ON torneo.citta = partecipa.citta
JOIN giocatore ON partecipa.classifica = giocatore.classifica
WHERE torneo.cogn_vincitore = giocatore.cognome
AND giocatore.classifica = 1;
```

```
+---------------+------------+-------+----------+
| Vittorie 2015 | classifica | nome  | cognome  |
+---------------+------------+-------+----------+
|             8 |          1 | Novak | Djokovic |
+---------------+------------+-------+----------+
1 row in set (0,00 sec)
```

## 3) VOGLIAMO CONOSCERE NOME E COGNOME DEI GIOCATORI CON WILSON COME SPONSOR TECNICO

```
SELECT giocatore.nome , giocatore.cognome
FROM giocatore JOIN supportato ON giocatore.classifica =
supportato.classifica
JOIN sponsor_tec ON supportato.marchio_t = sponsor_tec.marchio_t
WHERE sponsor_tec.marchio_t = "Wilson";
```

```
+----------+--------------+
| nome     | cognome      |
+----------+--------------+
| Andy     | Murray       |
| Tomas    | Berdych      |
| David    | Goffin       |
| Roberto  | BautistaAgut |
| Nick     | Kyrgios      |
| Pablo    | Cuevas       |
| Fabio    | Fognini      |
+----------+--------------+
24 rows in set (0,00 sec)
```

4) SI VUOLE CONOSCERE LA CIFRA RELATIVA ALL' APPORTO COMPLESSIOVO ANNUALE DELLO SPONSOR BNL.

```
SELECT sum(sponsor_maj.apporto) "TOT.APPORTO BNL"
FROM sponsor_maj JOIN finanziato ON
sponsor_maj.marchio_m = finanziato.marchio_m
JOIN torneo ON finanziato.citta = torneo.citta
WHERE sponsor_maj.marchio_m = "BNL" ;
```

```
+-----------------+
| TOT.APPORTO BNL |
+-----------------+
|         5789600 |
+-----------------+
1 row in set (0.02 sec)
```

5) VOGLIAMO NOME ,COGNOME E NAZIONE , DI SPETTATORI NON RUSSI AL TORNEO DI MOSCA

```
SELECT spettatore.nome, spettatore.cognome, spettatore.nazione
FROM spettatore JOIN biglietto ON spettatore.cod_f =
biglietto.cod_f
JOIN stadio ON biglietto.nome_st = stadio.nome_st
JOIN torneo ON stadio.citta = torneo.citta
WHERE torneo.citta = "Mosca"
AND spettatore.nazione != "RUS";
```

```
+-------------+------------+---------+
| nome        | cognome    | nazione |
+-------------+------------+---------+
| Roy         | Sanchez    | ENG     |
| Louis       | Marshall   | SWI     |
| Russell     | Owens      | MEX     |
| Catherine   | Medina     | CHN     |
| Jesse       | Kelley     | MEX     |
| Melissa     | Alexander  | JPN     |
| Elizabeth   | Ford       | SWI     |
| Aaron       | Andrews    | NOR     |
| Eric        | Diaz       | GER     |
| Mark        | Phillips   | CHN     |
| Douglas     | Woods      | GER     |
```

```
| Janice      | Freeman    | AUT     |
| Helen       | Long       | AUT     |
| Anne        | Campbell   | JPN     |
| Kimberly    | Reynolds   | CHN     |
| Pamela      | Dunn       | MEX     |
| Linda       | Knight     | JPN     |
| Theresa     | Perkins    | AUT     |
+-------------+------------+---------+
1933 rows in set (0.01 sec)
```

## 6) SELEZIONARE IL NOME DEL TORNEO CHE HA VENDUTO PIU' BIGLIETTI IN STAGIONE

```
SELECT COUNT(*) "biglietti Venduti" , torneo.nome_t
FROM biglietto JOIN stadio ON biglietto.nome_st = stadio.nome_st
JOIN torneo ON stadio.citta = torneo.citta
WHERE torneo.data_i BETWEEN '2015-01-1'
AND '2015-12-28'
GROUP BY (torneo.citta)
ORDER BY COUNT(*) DESC LIMIT 1;
```

```
+-------------------+------------+
| biglietti Venduti | nome_t     |
+-------------------+------------+
|              5064 | Paris-tour |
+-------------------+------------+
1 row in set (0,08 sec)
```

## 7) SELEZIONARE I 10 TORNEI SU CUI INVESTIRE MAGGIORMENTE IL PROSSIMO ANNO, ANALIZZANDO I RICAVI DEI BIGLIETTI VENDUTI , ED ORDINANDOLI IN MODO DECRESCENTE.

```
SELECT
COUNT(id_b) "biglietti Venduti" , torneo.nome_t ,
SUM(biglietto.prezzo)
FROM biglietto JOIN stadio ON biglietto.nome_st = stadio.nome_st
JOIN torneo ON stadio.citta = torneo.citta
WHERE torneo.data_i BETWEEN '2015-01-1'
AND '2015-12-28'
GROUP BY (torneo.citta)
ORDER BY COUNT(id_b) DESC LIMIT 10;
```

```
+-------------------+---------------+-----------------------+
| biglietti Venduti | nome_t        | SUM(biglietto.prezzo) |
+-------------------+---------------+-----------------------+
|              5064 | Paris-tour    |                615240 |
|              4982 | Milano-master |                601200 |
|              4933 | Dublino-cup   |                589170 |
|              4930 | Londra-ace    |                587850 |
|              4928 | NewYork-ace   |                592740 |
|              4923 | Basilea-tennis|                588600 |
|              4913 | Atene-master  |                593160 |
|              4858 | Miami-master  |                587730 |
|              4854 | Madrid-tour   |                579390 |
|              4839 | Berlino-tennis|                580320 |
+-------------------+---------------+-----------------------+
10 rows in set (0,28 sec)
```

## 8) ESTENDERE L' ANLISI PRECEDENTE ALLE 5 NAZIONI PIU' REDDITIZIE DEL CIRCUITO ATP.

```
SELECT
COUNT(id_b) "Tot Biglietti Venduti" , torneo.nazione ,
SUM(biglietto.prezzo) "Guadagno per Nazione"
FROM biglietto JOIN stadio ON biglietto.nome_st = stadio.nome_st
JOIN torneo ON stadio.citta = torneo.citta
WHERE torneo.data_i BETWEEN '2015-01-1'
AND '2015-12-28'
GROUP BY (torneo.nazione)
ORDER BY COUNT(id_b) DESC LIMIT 5;
```

```
+----------------------+---------+----------------------+
| Tot Biglietti Venduti | nazione | Guadagno per Nazione |
+----------------------+---------+----------------------+
|                28791 | ITA     |              3451830 |
|                20341 | ESP     |              2438640 |
|                20282 | CHN     |              2438910 |
|                20209 | USA     |              2430930 |
|                19869 | FRA     |              2376450 |
+----------------------+---------+----------------------+
5 rows in set (0,39 sec)
```

## 9) SELEZIONARE I 5 TORNEI PIU' ABMITI DELL ANNO CIOE' QUELLI CON P_MONEY MAGGIORE.

```
SELECT torneo.nome_t, torneo.nazione,torneo.citta ,torneo.p_money
FROM torneo
ORDER BY torneo.p_money DESC LIMIT 5;
```

```
+----------------+---------+----------+---------+
| nome_t         | nazione | citta    | p_money |
+----------------+---------+----------+---------+
| Krakowia-world | POL     | Krakowia | 4996325 |
| Doha-master    | QTA     | Doha     | 4965629 |
| Montreal-world | CAN     | Montreal | 4938602 |
| Vicenza-tour   | ITA     | Vicenza  | 4906683 |
| Baotou-tennis  | CHN     | Baotou   | 4891190 |
+----------------+---------+----------+---------+
5 rows in set (0,01 sec)
```

10) GLI SPONSOR CHE FINANZIANO I TORNEI VOGLIONO AVERE UN ANALISI CHE RESTITUISCA NOME, COGNOME, ETA' E TITOLI , DEI 5 GIOVANI PIU' VINCENTI DELL' ANNO PER INDIRIZZARE SU DI LORO LE CAMPAGNE PUBBLICITARIE.

```
SELECT giocatore.nome , giocatore.cognome ,
giocatore.eta ,giocatore.titoli
FROM torneo JOIN partecipa ON partecipa.citta = torneo.citta
JOIN giocatore ON giocatore.classifica = partecipa.classifica
WHERE giocatore.cognome = torneo.cogn_vincitore
AND giocatore.eta <= 23
GROUP BY (giocatore.nome)
ORDER BY (titoli) DESC;
```

```
+-----------+---------+-----+--------+
| nome      | cognome | eta | titoli |
+-----------+---------+-----+--------+
| Borna     | Coric   |  19 |     10 |
| Alexander | Zverev  |  18 |     10 |
| Dominic   | Thiem   |  22 |      7 |
| Denis     | Kudla   |  23 |      7 |
| Jiri      | Vesely  |  22 |      6 |
+-----------+---------+-----+--------+
5 rows in set (0,08 sec)
```

11) AI FINI DELL' ASSEGNAZIONE DEL PREMIO MIGLIOR GIOVANE, SELEZIONARE NOME, COGNOME, ETA, DEL PIU' GIOVANE VINCITORE DI ALMENO 1 TORNEO NEL 2015.

```
SELECT giocatore.nome , giocatore.cognome ,
giocatore.eta ,giocatore.titoli
FROM torneo JOIN partecipa ON partecipa.citta = torneo.citta
JOIN giocatore ON giocatore.classifica = partecipa.classifica
WHERE giocatore.cognome = torneo.cogn_vincitore
ORDER BY (eta) LIMIT 1;
```

```
+-----------+---------+-----+--------+
| nome      | cognome | eta | titoli |
+-----------+---------+-----+--------+
| Alexander | Zverev  |  18 |     10 |
+-----------+---------+-----+--------+
1 row in set (0,00 sec)
```

12) AI FINI DELL ASSEGNAZIONE DEL PREMIO MIGLIOR OVER 30 , SELEZIONARE NOME, COGNOME, ETA' DEL VINCITORE PIU' VECCHIO DI ALMENO 1 TORNEO NEL 2015.

```
SELECT giocatore.nome , giocatore.cognome ,
giocatore.eta ,giocatore.titoli
FROM torneo JOIN partecipa ON partecipa.citta = torneo.citta
JOIN giocatore ON giocatore.classifica = partecipa.classifica
WHERE giocatore.cognome = torneo.cogn_vincitore
ORDER BY (eta) DESC LIMIT 1;
```

```
+-----------+---------+-----+--------+
| nome      | cognome | eta | titoli |
+-----------+---------+-----+--------+
| Alexander | Zverev  |  18 |     10 |
+-----------+---------+-----+--------+
1 row in set (0,00 sec)
```

13) VOGLIAMO SCOPRIRE CHI E' SATO IL GIOCATORE PIU' FORTUNATO DELLA STAGIONE IN TERMINI DI INCASSI DERIVANTI DA VITTORIE.

```
SELECT giocatore.nome ,giocatore.cognome, SUM(torneo.p_money)
FROM torneo JOIN partecipa ON partecipa.citta = torneo.citta
JOIN giocatore ON giocatore.classifica = partecipa.classifica
WHERE giocatore.cognome = torneo.cogn_vincitore
GROUP BY (giocatore.nome)
ORDER BY (SUM(torneo.p_money)) DESC LIMIT 1 ;
```

```
+-------+---------+---------------------+
| nome  | cognome | SUM(torneo.p_money) |
+-------+---------+---------------------+
| Borna | Coric   |            34303076 |
+-------+---------+---------------------+
1 row in set (0,02 sec)
```

**14)** COME TUTTI , L' ATP E' IN CRISI. NECESSITIA QUINDI DI UNA PICCOLA ENTRATA EXTRA, RICAVABILE IMPONENDO UNA TASSA DEL 20% SUI GUADAGNI DERIVANTI DA VITTORIE AI GIOCATORI. SI MOSTRI IL LORDO, L' AMMONTARE DELLA TASSA ED IL NETTO PER OGNI GIOCATORE.

```
SELECT giocatore.nome ,giocatore.cognome, SUM(torneo.p_money),
((SUM(torneo.p_money) * 20 )/100) AS "TASSA",
(SUM(torneo.p_money) - ((SUM(torneo.p_money) * 20 )/100)) AS
"NETTO"
FROM torneo JOIN partecipa ON partecipa.citta = torneo.citta
JOIN giocatore ON giocatore.classifica = partecipa.classifica
WHERE giocatore.cognome = torneo.cogn_vincitore
GROUP BY (giocatore.nome)
ORDER BY (SUM(torneo.p_money)) DESC ;
```

| nome | cognome | SUM(torneo.p_money) | TASSA | NETTO |
|---|---|---|---|---|
| Borna | Coric | 34303076 | 6860615.2000 | 27442460.8000 |
| Milos | Raonic | 29290206 | 5858041.2000 | 23432164.8000 |
| Novak | Djokovic | 25310038 | 5062007.6000 | 20248030.4000 |
| Alexander | Zverev | 21504379 | 4300875.8000 | 17203503.2000 |
| Andy | Murray | 20846470 | 4169294.0000 | 16677176.0000 |
| Stan | Wawrinka | 20176179 | 4035235.8000 | 16140943.2000 |
| Denis | Kudla | 20114910 | 4022982.0000 | 16091928.0000 |
| Rafael | Nadal | 19875909 | 3975181.8000 | 15900727.2000 |
| Jiri | Vesely | 19255212 | 3851042.4000 | 15404169.6000 |
| Tomas | Berdych | 15864439 | 3172887.8000 | 12691551.2000 |
| Dominic | Thiem | 14485032 | 2897006.4000 | 11588025.6000 |
| David | Ferrer | 12678460 | 2535692.0000 | 10142768.0000 |
| Roger | Federer | 11342600 | 2268520.0000 | 9074080.0000 |
| Grigor | Dimitrov | 3357015 | 671403.0000 | 2685612.0000 |
| Kei | Nishikori | 2576857 | 515371.4000 | 2061485.6000 |

15 rows in set (0,02 sec)

**15)** SELEZIONARE ELENCANDO NOME E COGNOME DEL #1 AL MONDO, OLTRE AL NOME E PRYCE MONEY DEL TORNEO , COSI DA DARE A LUI UN ADEGUATA STATISTICA DEI PUNTI DA DIFENDERE NEL 2016.

```
SELECT giocatore.nome ,giocatore.cognome, torneo.nome_t,
torneo.p_money
FROM torneo JOIN partecipa ON partecipa.citta = torneo.citta
JOIN giocatore ON giocatore.classifica = partecipa.classifica
WHERE giocatore.cognome = torneo.cogn_vincitore
AND giocatore.classifica = 1;
```

| nome | cognome | nome_t | p_money |
|---|---|---|---|
| Novak | Djokovic | Baotou-tennis | 4891190 |
| Novak | Djokovic | Beijing-cup | 1356706 |
| Novak | Djokovic | Cincinnati-ace | 335257 |
| Novak | Djokovic | Launcheston-cup | 3960449 |
| Novak | Djokovic | Marrakech-master | 2621276 |
| Novak | Djokovic | Oslo-cup | 3868090 |
| Novak | Djokovic | Pires-master | 4343575 |
| Novak | Djokovic | Pozan-cup | 3933495 |

8 rows in set (0,00 sec)

## 16) CONTARE L' AMMONTARE DI SPETTATORI DI UNA DETERMINATA NAZIONALITA' PRESENTI IN STAGIONE.

```
SELECT nazione , COUNT(nazione) "Tot Italiani"
FROM spettatore
WHERE nazione = "ITA";
```

```
+---------+---------------+
| nazione | Tot Italiani |
+---------+---------------+
| ITA     |          6209 |
+---------+---------------+
1 row in set (0,22 sec)
```

```
+----+-------------+------------+------+---------------+------+---------+------+-------+-------------+
| id | select_type | table      | type | possible_keys | key  | key_len | ref  | rows  | Extra       |
+----+-------------+------------+------+---------------+------+---------+------+-------+-------------+
|  1 | SIMPLE      | spettatore | ALL  | NULL          | NULL | NULL    | NULL | 95712 | Using where |
+----+-------------+------------+------+---------------+------+---------+------+-------+-------------+
1 row in set (0,00 sec)
```

DOPO L INSERIMENTO DI UN INDICE I2 SU SPETTATORE(nazione)

```
+---------+---------------+
| nazione | Tot Italiani |
+---------+---------------+
| ITA     |          6209 |
+---------+---------------+
1 row in set (0,03 sec)
```

```
+----+-------------+------------+------+---------------+------+---------+-------+------+--------------------------+
| id | select_type | table      | type | possible_keys | key  | key_len | ref   | rows | Extra                    |
+----+-------------+------------+------+---------------+------+---------+-------+------+--------------------------+
|  1 | SIMPLE      | spettatore | ref  | i2            | i2   | 5       | const | 6208 | Using where; Using index |
+----+-------------+------------+------+---------------+------+---------+-------+------+--------------------------+
1 row in set (0,00 sec)
```

## 17) SEMPRE PER FINI PROMOZIONALI ELENCARE LA LISTA DEI 3 POPOLI PIU' PRESENTI IN STAGIONE NEL CIRCUITO, SPECIFICANDONE LA NAZIONE E L' AMMONTARE PER OGNUNA DI ESSE.

```
SELECT nazione , count(nazione) "Tot per nazione"
FROM spettatore
GROUP BY nazione
ORDER BY COUNT(nazione) DESC LIMIT 3;
```

```
+---------+-----------------+
| nazione | Tot per nazione |
+---------+-----------------+
| ENG     |            6391 |
| AUS     |            6337 |
| NOR     |            6315 |
+---------+-----------------+
3 rows in set (0,06 sec)
```

18) SI VUOLE ANALIZZARE L AFFLUSSO AD UN TORNEO DI UN PUBBLICO FEMMINILE NON SUPERIORE I 30 ANNI , NEL TORNEO DI ROMA. IN PARTICOLARE VOGLIAMO ANALIZZARE LA CORRELAZIONE CHE ESISTE TRA, PREZZO DEL BIGLIETTO E FASCIA GIOVANE PUBBLICO SESSO FEMMINILE.

```
SELECT prezzo , COUNT(spettatore.sesso) "TOT biglietti per sesso
femminile"
FROM spettatore JOIN biglietto ON
spettatore.cod_f = biglietto.cod_f
JOIN stadio ON biglietto.nome_st = stadio.nome_st
JOIN torneo ON stadio.citta = torneo.citta
WHERE spettatore.sesso = "F"
AND spettatore.eta <= 30
AND torneo.citta = "Roma"
GROUP BY biglietto.prezzo
ORDER BY COUNT(spettatore.sesso);
```

| prezzo | TOT biglietti per sesso femminile |
|---|---|
| 210 | 64 |
| 90 | 66 |
| 120 | 67 |
| 30 | 69 |
| 150 | 72 |
| 60 | 77 |
| 180 | 84 |

7 rows in set (0,02 sec)

19) VOGLIAMO LO SPONSOR PIU' FACOLTOSO ESCLUSI I MAJOR SPONSOR PER ECCELLENZA BNL ed Fly-Emirates.

```
SELECT sponsor_maj.marchio_m, SUM(sponsor_maj.apporto)
"TOT.APPORTO BNL"
FROM sponsor_maj JOIN finanziato ON
sponsor_maj.marchio_m = finanziato.marchio_m
JOIN torneo ON finanziato.citta = torneo.citta
WHERE sponsor_maj.marchio_m != "BNL"
AND  sponsor_maj.marchio_m != "Fly-Emirates"
GROUP BY (sponsor_maj.marchio_m)
ORDER BY SUM(sponsor_maj.apporto) DESC LIMIT 1;
```

| marchio_m | TOT.APPORTO BNL |
|---|---|
| Porche | 55284208 |

1 row in set (0,05 sec)

20) AI FINI DELLA CORRETTA ASSEGNAZIONE DEL PREMIO FEDELTA', VOGLIAMO VISUALIZZARE LA LISTA DEI PRIMI 100 SPETTATORI OVER 60 PIU' PRESENTI NEL CIRCUITO.

```
SELECT spettatore.nome ,
spettatore.eta,spettatore.cognome,count(id_b) AS "Acquistati"
FROM spettatore JOIN biglietto ON
spettatore.cod_f = biglietto.cod_f
WHERE eta > 60
GROUP BY biglietto.cod_f
ORDER BY count(id_b) DESC LIMIT 100;
```

| nome | eta | cognome | Acquistati |
|---|---|---|---|
| Maria | 77 | Foster | 9 |
| Randy | 61 | Walker | 9 |
| Steven | 74 | Murray | 7 |
| Annie | 64 | Fisher | 7 |
| Jane | 78 | Hanson | 7 |
| Carolyn | 78 | Reynolds | 7 |
| Jeremy | 68 | Simmons | 7 |
| Jimmy | 63 | Kennedy | 7 |

100 rows in set (0,17 sec)

21) PER FINI STATISTICI VOGLIAMO SAPERE QUALI SONO I BRAND PIU' INFLUENTI IN UN DETERMINATO PAESE, CIOE' QUEI BRAND CHE SPINGONO LE PERSONE AD ANDARE AL TORNEO, PER ACCAPARRARSI GADGET ecc…

```
SELECT
COUNT(id_b) "presenti" , sponsor_maj.marchio_m
FROM spettatore JOIN biglietto ON spettatore.cod_f =
biglietto.cod_f
JOIN stadio ON biglietto.nome_st = stadio.nome_st
JOIN torneo ON stadio.citta = torneo.citta
JOIN finanziato ON torneo.citta = finanziato.citta
JOIN sponsor_maj ON finanziato.marchio_m = sponsor_maj.marchio_m
WHERE torneo.nazione = "ENG"
AND spettatore.nazione = "ENG"
AND sponsor_maj.marchio_m != "BNL"
AND sponsor_maj.marchio_m != "Fly-Emirates"
GROUP BY sponsor_maj.marchio_m
ORDER BY COUNT(id_b);
```

| id | select_type | table | type | possible_keys | key | key_len | ref | rows | Extra |
|---|---|---|---|---|---|---|---|---|---|
| 1 | SIMPLE | torneo | ALL | PRIMARY | NULL | NULL | NULL | 100 | Using where; Using temporary; Using filesort |
| 1 | SIMPLE | stadio | ref | PRIMARY,citta | citta | 22 | atp.torneo.citta | 1 | Using index |
| 1 | SIMPLE | finanziato | ref | marchio_m,citta | citta | 22 | atp.torneo.citta | 26 | Using where |
| 1 | SIMPLE | sponsor_maj | eq_ref | PRIMARY,i2 | PRIMARY | 22 | atp.finanziato.marchio_m | 1 | Using index |
| 1 | SIMPLE | biglietto | ref | cod_f,nome_st | nome_st | 22 | atp.stadio.nome_st | 332 | NULL |
| 1 | SIMPLE | spettatore | eq_ref | PRIMARY | PRIMARY | 12 | atp.biglietto.cod_f | 1 | Using where |

6 rows in set (0,00 sec)

| presenti | marchio_m |
|---|---|
| 1459 | Polo ASSN |
| 1441 | Algida |
| 1341 | Generali |
| 133 | Asics |
| 133 | TelCEL |
| 132 | Richo |
| 132 | Globo |

96 rows in set (1,50 sec)

**CREIAMO UN INDICE SULLA NAZIONE**

```
create index i1 on spettatore(nazione);
```

| id | select_type | table | type | possible_keys | key | key_len | ref | rows | Extra |
|---|---|---|---|---|---|---|---|---|---|
| 1 | SIMPLE | spettatore | ref | PRIMARY,i1 | i1 | 5 | const | 6390 | Using where; Using index; Using temporary; Using filesort |
| 1 | SIMPLE | biglietto | ref | cod_f,nome_st | cod_f | 12 | atp.spettatore.cod_f | 1 | NULL |
| 1 | SIMPLE | stadio | eq_ref | PRIMARY,citta | PRIMARY | 22 | atp.biglietto.nome_st | 1 | NULL |
| 1 | SIMPLE | torneo | eq_ref | PRIMARY | PRIMARY | 22 | atp.stadio.citta | 1 | Using where |
| 1 | SIMPLE | finanziato | ref | marchio_m,citta | citta | 22 | atp.stadio.citta | 26 | Using where |
| 1 | SIMPLE | sponsor_maj | eq_ref | PRIMARY,i2 | PRIMARY | 22 | atp.finanziato.marchio_m | 1 | Using index |

| presenti | marchio_m |
|---|---|
| 1459 | Polo ASSN |
| 1441 | Algida |
| 1341 | Generali |
| 133 | TelCEL |
| 133 | Asics |
| 132 | Richo |
| 132 | Globo |

96 rows in set (0,16 sec)

Possiamo notare dall' explain che il numero di righe considerato nella query ottimizzata, è pari ad 1/8 di quello utilizzato dalla prima.

## 22) DIRE QUAL  E' LO SPONSOR TECNICO CHE SUPPORTA PIU' GIOCATORI NEL CIRCUITO

```
SELECT sponsor_tec.marchio_t , COUNT(sponsor_tec.marchio_t)
FROM sponsor_tec JOIN supportato ON sponsor_tec.marchio_t =
supportato.marchio_t
JOIN giocatore ON supportato.classifica = giocatore.classifica
GROUP BY sponsor_tec.marchio_t
ORDER BY COUNT(sponsor_tec.marchio_t) DESC;
```

| marchio_t | COUNT(sponsor_tec.marchio_t) |
|---|---|
| Meemm | 31 |
| Lotto | 31 |
| Devpoint | 27 |
| Artengo | 27 |
| Realmix | 27 |
| Dunlop | 25 |
| Head | 25 |
| Adidas | 24 |
| Yacero | 24 |
| Wilson | 24 |
| Prince | 21 |
| Tecnofibre | 20 |
| Ooba | 20 |
| Babolat | 20 |
| Kazio | 19 |
| Dablist | 19 |
| Nike | 19 |
| Roodel | 18 |
| Lacoste | 18 |
| Flipbug | 17 |
| Quinu | 16 |
| Gigabox | 15 |
| Feedfire | 13 |

23 rows in set (0,02 sec)

## 23) CONTARE GLI SPETTATORI PIU ANZIANI DEL 2015

```
SELECT COUNT(spettatore.cod_f) "I più anziani" ,spettatore.eta
FROM spettatore
WHERE spettatore.eta = (
     SELECT max(spettatore.eta)
     FROM spettatore);
```

| I più anziani | eta |
|---|---|
| 1612 | 80 |

1 row in set (0,07 sec)

/* una versione più elegante e leggermente più efficiente per evitare la nidificazione */

```
set  @test = (SELECT max(eta) FROM spettatore)

SELECT COUNT(spettatore.cod_f) "I più anziani" , spettatore.eta
FROM spettatore
WHERE spettatore.eta  = @test;
```

| I più anziani | eta |
|---|---|
| 1612 | 80 |

1 row in set (0,04 sec)

## 24) CONTARE GLI SPETTATORI CON MASSIMO 21 ANNI NEL 2015

```
SELECT count( spettatore.eta) "Giovani"
FROM spettatore
WHERE spettatore.eta < 22;
```

```
+----------+
| Giovani  |
+----------+
|     6313 |
+----------+
1 row in set (0,19 sec)
```

## 25) VOGLIAMO AVERE IL DATO RIGUARDANTE IL BIGLIETTO PIU' APPREZZATO DAI GIOVANI, COSI DA RIPROPORLO NELLA STAGIONE 2016

```
SELECT
count(biglietto.id_b) "Venduti" , biglietto.prezzo
FROM biglietto JOIN spettatore ON
biglietto.cod_f = spettatore.cod_f
WHERE spettatore.eta < 22
GROUP BY biglietto.prezzo
ORDER BY COUNT(biglietto.id_b) desc;
```

```
+----+-------------+-----------+------+---------------+-------+---------+---------------------+-------+----------------------------------------------+
| id | select_type | table     | type | possible_keys | key   | key_len | ref                 | rows  | Extra                                        |
+----+-------------+-----------+------+---------------+-------+---------+---------------------+-------+----------------------------------------------+
|  1 | SIMPLE      | spettatore | ALL  | PRIMARY       | NULL  | NULL    | NULL                | 95712 | Using where; Using temporary; Using filesort |
|  1 | SIMPLE      | biglietto | ref  | cod_f         | cod_f | 12      | atp.spettatore.cod_f |     1 | NULL                                         |
+----+-------------+-----------+------+---------------+-------+---------+---------------------+-------+----------------------------------------------+
2 rows in set (0,00 sec)
```

```
+---------+--------+
| Venduti | prezzo |
+---------+--------+
|    2342 |    150 |
|    2331 |     60 |
|    2320 |     90 |
|    2274 |    120 |
|    2224 |     30 |
|    2194 |    180 |
|    2170 |    210 |
+---------+--------+
7 rows in set (0,24 sec)
```

```
create index i2 on spettatore(eta); /* lieve miglioramento */
```

```
+----+-------------+-----------+-------+---------------+------+---------+---------------------+------+----------------------------------------------------+
| id | select_type | table     | type  | possible_keys | key  | key_len | ref                 | rows | Extra                                              |
+----+-------------+-----------+-------+---------------+------+---------+---------------------+------+----------------------------------------------------+
|  1 | SIMPLE      | spettatore | range | PRIMARY,i2    | i2   | 1       | NULL                | 6312 | Using where; Using index; Using temporary; Using filesort |
|  1 | SIMPLE      | biglietto | ref   | cod_f         | cod_f | 12      | atp.spettatore.cod_f |    1 | NULL                                               |
+----+-------------+-----------+-------+---------------+------+---------+---------------------+------+----------------------------------------------------+
2 rows in set (0,00 sec)
```

```
+---------+--------+
| Venduti | prezzo |
+---------+--------+
|    2342 |    150 |
|    2331 |     60 |
|    2320 |     90 |
|    2274 |    120 |
|    2224 |     30 |
|    2194 |    180 |
|    2170 |    210 |
+---------+--------+
7 rows in set (0,07 sec)
```

## 26) SI VUOLE PREMIARE IL MIGLIOR ARBITRO DELLA STAGIONE IN BASE AL NUMERO DI FINALI ARBITRATE. INOLTRE PER OGNI FINALE VERRA DATO UN COMPENSO ALL' ARBITRO. CALCOLARE IL TOTALE SUL MIGLIOR ARBITRO

```
SELECT  arbitri.nome , arbitri.cognome ,arbitri.matricola_ar,
COUNT( arbitri.matricola_ar) "Finali arbitrate",
(COUNT(arbitri.matricola_ar)* 25000) AS "Compenso_lordo_arbitro#1"
FROM  arbitri JOIN torneo ON
arbitri.matricola_ar = torneo.matricola_ar
GROUP BY arbitri.matricola_ar
ORDER BY COUNT( arbitri.matricola_ar) DESC LIMIT 1;
```

```
+---------+---------+--------------+------------------+---------------------------+
| nome    | cognome | matricola_ar | Finali arbitrate | Compenso_lordo_arbitro#1 |
+---------+---------+--------------+------------------+---------------------------+
| Rebecca | Ortiz   | 10A          |                3 |                     75000 |
+---------+---------+--------------+------------------+---------------------------+
1 row in set (0,00 sec)
```

## 27) VOGLIAMO SAPERE QUANTI SPETTATORI HANNO ASSISTITO A TORNEI DI FEDERER

```
explain SELECT COUNT(spettatore.cod_f) "ROGERIANI"
FROM spettatore JOIN biglietto ON
spettatore.cod_f = biglietto.cod_f
JOIN stadio ON biglietto.nome_st = stadio.nome_st
JOIN torneo ON stadio.citta = torneo.citta
WHERE torneo.cogn_vincitore = "Federer";
```

```
+-----------+
| ROGERIANI |
+-----------+
|     13434 |
+-----------+
1 row in set (0,20 sec)
```

```
+----+-------------+-----------+--------+---------------+---------+---------+---------------------+------+-------------+
| id | select_type | table     | type   | possible_keys | key     | key_len | ref                 | rows | Extra       |
+----+-------------+-----------+--------+---------------+---------+---------+---------------------+------+-------------+
|  1 | SIMPLE      | torneo    | ALL    | PRIMARY       | NULL    | NULL    | NULL                |  100 | Using where |
|  1 | SIMPLE      | stadio    | ref    | PRIMARY,citta | citta   | 22      | atp.torneo.citta    |    1 | Using index |
|  1 | SIMPLE      | biglietto | ref    | cod_f,nome_st | nome_st | 22      | atp.stadio.nome_st  |  332 | NULL        |
|  1 | SIMPLE      | spettatore| eq_ref | PRIMARY       | PRIMARY | 12      | atp.biglietto.cod_f |    1 | Using index |
+----+-------------+-----------+--------+---------------+---------+---------+---------------------+------+-------------+
4 rows in set (0,00 sec)
```

```
Create index i3 on torneo(cogn_vincitore);
```

```
+----+-------------+-----------+--------+---------------+---------+---------+---------------------+------+--------------------------+
| id | select_type | table     | type   | possible_keys | key     | key_len | ref                 | rows | Extra                    |
+----+-------------+-----------+--------+---------------+---------+---------+---------------------+------+--------------------------+
|  1 | SIMPLE      | torneo    | ref    | PRIMARY,i3    | i3      | 17      | const               |    5 | Using where; Using index |
|  1 | SIMPLE      | stadio    | ref    | PRIMARY,citta | citta   | 22      | atp.torneo.citta    |    1 | Using index              |
|  1 | SIMPLE      | biglietto | ref    | cod_f,nome_st | nome_st | 22      | atp.stadio.nome_st  |  332 | NULL                     |
|  1 | SIMPLE      | spettatore| eq_ref | PRIMARY       | PRIMARY | 12      | atp.biglietto.cod_f |    1 | Using index              |
+----+-------------+-----------+--------+---------------+---------+---------+---------------------+------+--------------------------+
4 rows in set (0,00 sec)
```

```
+-----------+
| ROGERIANI |
+-----------+
|     13434 |
+-----------+
1 row in set (0,03 sec)
```

## 28) MOSTRARE LA TV CHE HA TRASMESSO PIU' TORNEI IN HD NEL 2015

```
SELECT
COUNT(tv.canale) "Trasmissioni" , tv.canale
FROM tv JOIN trasmesso ON tv.canale = trasmesso.canale
JOIN torneo ON trasmesso.citta = torneo.citta
WHERE trasmesso.canale = tv.canale AND tv.hd = 1
GROUP BY tv.canale
ORDER BY COUNT(tv.canale) DESC;
```

```
+--------------+-------------+
| Trasmissioni | canale      |
+--------------+-------------+
|          115 | dutch2      |
|          112 | supertennis |
|          112 | bigtv1      |
|           87 | canal6      |
|           85 | canal2      |
|           83 | srf1        |
+--------------+-------------+
30 rows in set (0,05 sec)
```

## 29) VOGLIAMO UNA STATISTICA SUGLI SPONSOR TECNICI CHE SPONSORIZZAVANO I GIOCATORI NELLE FINALI 2015, IN MODO DA FORNIRE UNA PANORAMICA SUI MIGLIORI SPONSOR TECNICI.

```
SELECT COUNT(sponsor_tec.marchio_t) , sponsor_tec.marchio_t AS
sponsor_in_finale_da_vincitori
FROM sponsor_tec JOIN supportato ON
sponsor_tec.marchio_t =supportato.marchio_t
jOIN giocatore ON  supportato.classifica = giocatore.classifica
JOIN partecipa ON giocatore.classifica = partecipa.classifica
JOIN torneo ON partecipa.citta = torneo.citta
WHERE giocatore.cognome = torneo.cogn_vincitore
GROUP BY sponsor_tec.marchio_t
ORDER BY COUNT(sponsor_tec.marchio_t) DESC;
```

```
+-----------------------------+-------------------------------+
| COUNT(sponsor_tec.marchio_t) | sponsor_in_finale_da_vincitori |
+-----------------------------+-------------------------------+
|                          25 | Meemm                         |
|                          21 | Head                          |
|                          20 | Nike                          |
|                          19 | Yacero                        |
|                          18 | Artengo                       |
|                          17 | Prince                        |
|                          17 | Ooba                          |
|                          17 | Lotto                         |
|                          16 | Wilson                        |
|                          13 | Adidas                        |
|                          12 | Dunlop                        |
|                          10 | Devpoint                      |
|                           9 | Kazio                         |
|                           8 | Roodel                        |
|                           8 | Realmix                       |
|                           7 | Dablist                       |
|                           6 | Lacoste                       |
|                           2 | Babolat                       |
+-----------------------------+-------------------------------+
18 rows in set (0,02 sec)
```

# QUERY ALGEBRA RELAZIONALE

### 24) CONTARE GLI SPETTATORI CON MASSIMO 21 ANNI NEL 2015

```
SELECT count( spettatore.eta) "Giovani"
FROM spettatore
WHERE spettatore.eta < 22;
```

```
π count(eta)"Giovani"
σ( eta < 22 ( spettatore ))
```

### 5) VOGLIAMO NOME ,COGNOME E NAZIONE , DI SPETTATORI NON RUSSI AL TORNEO DI MOSCA

```
SELECT spettatore.nome, spettatore.cognome, spettatore.nazione
FROM spettatore JOIN biglietto ON spettatore.cod_f =
biglietto.cod_f
JOIN stadio ON biglietto.nome_st = stadio.nome_st
JOIN torneo ON stadio.citta = torneo.citta
WHERE torneo.citta = "Mosca"
AND spettatore.nazione != "RUS";
```

```
π nome, cognome , nazione
σ(spettatore.cod_f = biglietto.cod_f,
  biglietto.nome_st = stadio.nome_st,
  stadio.citta = torneo.citta,
  torneo.citta = “Mosca”,
  spettatore.nazione != “RUS”
     ( spettatore |X| biglietto |X| stadio |X| torneo ))
```

# VIEW

sono tabelle "virtuali" che condividono lo stesso spazio di memoria della tabella fisica alla quale fanno riferimento. Sono utili per quanto riguarda la presentazione del DB con l' esterno, ma possono essere pericolose in aggiornamento, perchè si rischia di modificare dati nella tabella originale.

### ViewTorneo

```
CREATE VIEW ViewTorneo AS
SELECT citta, nome_t , p_money , cogn_vincitore , nazione , data_i , data_f
FROM torneo;
```

### ViewVincitori

```
CREATE VIEW ViewVincitori AS
SELECT giocatore.classifica, torneo.cogn_vincitore, giocatore.nome,
giocatore.nazione
FROM torneo JOIN partecipa ON torneo.citta = partecipa.citta
JOIN giocatore ON partecipa.classifica = giocatore.classifica
WHERE torneo.cogn_vincitore = giocatore.cognome
GROUP BY giocatore.cognome
ORDER BY giocatore.classifica;
```

### ViewFinalisti

```
CREATE VIEW ViewFinalisti AS
SELECT  giocatore.classifica, torneo.cogn_finalista, giocatore.nome ,
giocatore.nazione
FROM torneo JOIN partecipa ON torneo.citta = partecipa.citta
JOIN giocatore ON partecipa.classifica = giocatore.classifica
WHERE torneo.cogn_finalista = giocatore.cognome
GROUP BY giocatore.cognome
ORDER BY giocatore.classifica;
```

### ViewStadio

```
CREATE VIEW ViewStadio AS
SELECT stadio.nome_st ,stadio.citta ,stadio.capienza
FROM stadio;
```

### ViewSponsorTec

```
CREATE VIEW ViewSponsorTec AS
SELECT sponsor_tec.marchio_t , sponsor_tec.p_iva , sponsor_tec.genere
FROM sponsor_tec;
```

# TRIGGER

### Trigger per il blocco dell' inserimento di spettatori minorenni.

```
DELIMITER $$

CREATE TRIGGER Alert_Minorenni
BEFORE INSERT ON spettatore
FOR EACH ROW BEGIN

DECLARE msg VARCHAR(255);

IF  ( NEW.eta < 18 ) THEN
     set msg = concat('Impossibile inserire nel DB un individuo
     minorenne, per analisi di mercato, nel rispetto dell articolo
     52.');
     signal sqlstate '45000' set message_text = msg;
END IF;

END $$

DELIMITER ;
```

### Trigger per rispettare la regola del "supporto massimo" per cui ogni giocatore non può superare un max di 3 sponsor tecnici.

```
DELIMITER $$

CREATE TRIGGER Sponsor
BEFORE INSERT ON supportato
FOR EACH ROW BEGIN

DECLARE msg VARCHAR(255);

SET @ACTUAL=(SELECT count(supportato.classifica) "num sponsor"
                FROM supportato
                WHERE supportato.classifica = NEW.classifica
               );

SET @MAX = 2;

IF  ( @ACTUAL > @MAX ) THEN
     set msg = concat('Troppi sponsor tecnici per il giocatore!);
     signal sqlstate '45000' set message_text = msg;
END IF;

END $$

DELIMITER ;
```

# SICUREZZA

Suddividiamo gli accessi al database , identificando 4 figure principali e garantendo loro i permessi adeguati allo svolgimento dell' attività lavorativa.

### ADMIN

livello che identifica le persone che gestiscono il database. Naturalmente essi avranno ogni tipo di permesso ed accesso.

```
GRANT all privileges
ON *.*
TO 'admin'@'localhost' identified by 'password' with GRANT option;
```

### ADDETTI ALLA MANUTENZIONE ED AGGIORNAMENTO DEL DB

livello che identifica coloro che, si occupano degli inserimenti, aggiornamenti e cancellazione dati nel database.

```
GRANT SELECT, INSERT, UPDATE, DELETE, privileges
ON Atp.*
To 'manutentore'@'esempio.com' identified by 'password';
```

### UTENTI

livello che identifica persone che hanno accesso parziale al database e con azioni limitate su di esso. Per motivi di sicurezza , come spesso accade nella realtà, agli utenti vengono rese visibili solo apposite viste del Db.

```
GRANT SELECT privileges
ON Atp.ViewTorneo, Atp.ViewVincitori, Atp.ViewFinalisti,
Atp.ViewStadio,Atp.ViewSponsorTec
TO 'userTest'@'esempio.com';
```

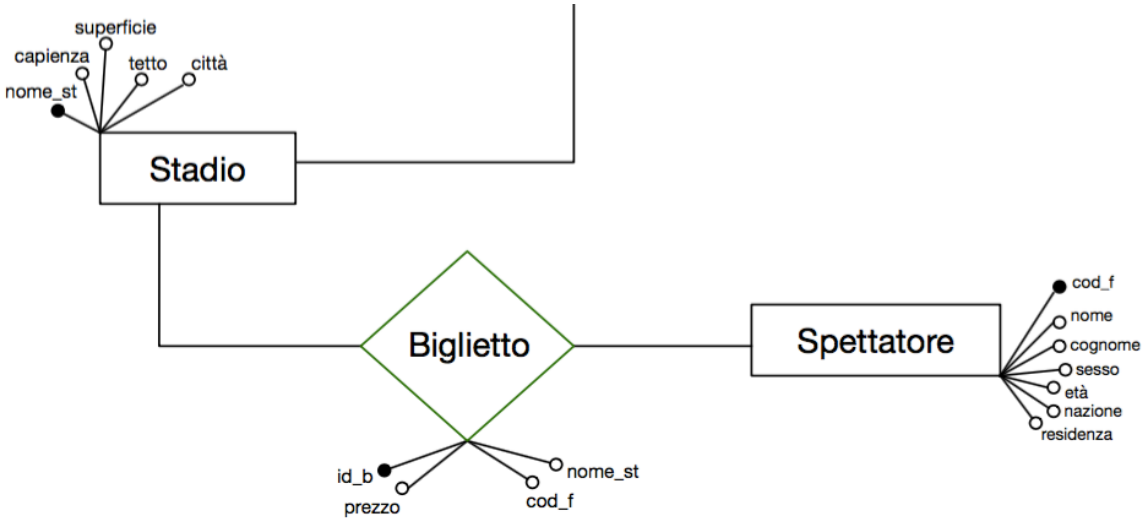
### ANALISTI & STAMPA AUTORIZZATA

persone che per fini statistici o giornalistici hanno bisogno , previa concessione dell Atp di accedere all' intero db, cosi da poter sottomettere le interrogazioni di loro interesse.

```
GRANT SELECT privileges
ON Atp.*
TO 'stampaEdAnalisti'@'esempio.com' identified by 'password' ;
```

# MONGODB

Per la creazione del nostro database in mongoDB , scegliamo di riprodurre un ramo del database atp in sql, con una quantità di dati molto simile, per cercare di operare confronti sui tempi di risposta dei due DB.




Riporteremo in mongoDB il seguente ramo generando nella collection spett ( spettatori ) 100.000 documents cosi strutturati.

```
{
      “_id":"Nx1lFRnOYoNm6G2A",
      "nome":"Amy",
      "cognome":"Greene",
      "sesso":"F",
      "eta":66,
      "nazione":"SWI",
      "residenza":"Milano",
      "biglietti": {
            "cod_b":"1B",
            "stadio":{
                  "nome":"Blois-One",
                  "tetto":"Yes",
                  "superficie":"cemento",
                  "citta":"Blois"
            },
            "prezzo":30
      }
},
```

adottiamo un id custom rappresentato dal codice fiscale.

notiamo che in ogni document della collection spett , troviamo il document innestato rappresentante il biglietto , che a sua volta ne presenta uno ulteriore , rappresentante lo stadio.

# INSERIMENTO DATI

- Per prima cosa creiamo il database app in mongoDB , con il semplice comando:

```
USE atp
```

questo comando seleziona un db esistente oppure, nel caso non sia presente lo crea e  lo seleziona per l utilizzo.

- Creiamo la collection spett (spettatori), che sintetizza il ramo scelto

```
db.createCollection("spett")
```

- Inseriamo i dati importandoli da un file spett.json precedentemente generato

```
sudo /Users/Framo/mongodb/mongodb-osx-x86_64-3.2.5/bin/mongoimport
--db atp --collection spett
--drop --file /Users/Framo/Desktop/Programming/Mongo_db/spett.json
```

```
2016-05-14T18:36:59.330+0200    connected to: localhost
2016-05-14T18:36:59.333+0200    dropping: atp.spett
2016-05-14T18:37:02.200+0200    imported 100000 documents
```

notiamo la velocità dell inserimento, pari a 2/3 secondi , tempi ottenibili in mysql , soltanto attraverso lettura ed importazione dati  "destrutturati", cioè senza la tipica sintassi dell' insert, ma leggendoli da file .txt , .csv ecc , con la procedura LoadData.

# CONFRONTO FIND CON QUERY SQL

## #24 CONTARE GLI SPETTATORI CON MASSIMO 21 ANNI NEL 2015

```
SELECT count( spettatore.eta) "Giovani"
FROM spettatore
WHERE spettatore.eta < 22;
```

```
+----------+
| Giovani |
+----------+
|    6313 |
+----------+
1 row in set (0,19 sec)
```

## #24 IN MONGODB

```
> db.spett.find({"eta": { $lt : 22 }}).count()
6323
```

```
> db.spett.find({"eta": { $lt : 22 }}).explain("executionStats")
```

```
"executionStats" : {
        "executionSuccess" : true,
        "nReturned" : 6323,
        "executionTimeMillis" : 54,
        "totalKeysExamined" : 0,
        "totalDocsExamined" : 100000,
        "executionStages" : {
                "stage" : "COLLSCAN",
                "filter" : {
                        "eta" : {
                                "$lt" : 22
                        }
                },
```

Analizzando la find con l apposito comando, abbiamo informazioni su piano d esecuzione uso di eventuali indici, nonché tempi d esecuzione effettivi e stimati. Proprio confrontando i tempi notiamo che mongoDB esegue la stessa query di mySQL in **0,053** sec contro i **0,190** sec.

## #16 CONTARE L AMMONTARE DI SPETTATORI DI UNA DETERMINATA NAZIONALITA' PRESENTI IN STAGIONE.

```
SELECT nazione , COUNT(nazione) "Tot Italiani"
FROM spettatore
WHERE nazione = "ITA";
```

```
+---------+--------------+
| nazione | Tot Italiani |
+---------+--------------+
| ITA     |         6209 |
+---------+--------------+
1 row in set (0,03 sec)
```

## #16 IN MONGODB

Ricordiamo che la query #16 sfrutta un indice su spettatore(nazione) per arrivare ad un tempo di 0.03 sec. Vediamo quindi l esecuzione della find in mongoDB sia con che senza indice per confrontare i tempi.

```
> db.spett.find({"nazione": "ITA" }).count()
6227
```

```
> db.spett.find({"nazione": "ITA" }).explain("executionStats")
```

i

```
"executionStats" : {
        "executionSuccess" : true,
        "nReturned" : 6227,
        "executionTimeMillis" : 59,
        "totalKeysExamined" : 0,
        "totalDocsExamined" : 100000,
        "executionStages" : {
                "stage" : "COLLSCAN",
                "filter" : {
                        "nazione" : {
                                "$eq" : "ITA"
                        }
                },
```

creiamo l' indice con l' apposita funzione. Se non specificato, il nome dell' indice sarà nome campo + tipo ordinamento( 1 , -1 ).

```
> db.spett.createIndex({"nazione" : 1 })
```

ed eseguiamo l explain:

```
> db.spett.find({"nazione": "ITA" }).explain("executionStats")
```

```
"queryPlanner" : {
        "plannerVersion" : 1,
        "namespace" : "atp.spett",
        "indexFilterSet" : false,
        "parsedQuery" : {
                "nazione" : {
                        "$eq" : "ITA"
                }
        },
        "winningPlan" : {
                "stage" : "FETCH",
                "inputStage" : {
                        "stage" : "IXSCAN",
                        "keyPattern" : {
                                "nazione" : 1
                        },
                        "indexName" : "nazione_1",
                        "isMultiKey" : false,
                        "isUnique" : false,
                        "isSparse" : false,
                        "isPartial" : false,
                        "indexVersion" : 1,
                        "direction" : "forward",
                        "indexBounds" : {
                                "nazione" : [
                                        "[\"ITA\", \"ITA\"]"
                                ]
                        }
                }
        },
        "rejectedPlans" : [ ]
},
"executionStats" : {
        "executionSuccess" : true,
        "nReturned" : 6227,
        "executionTimeMillis" : 9,
        "totalKeysExamined" : 6227,
        "totalDocsExamined" : 6227,
        "executionStages" : {
                "stage" : "FETCH",
                "nReturned" : 6227,
                "executionTimeMillisEstimate" : 0,
                "works" : 6228,
                "advanced" : 6227,
                "needTime" : 0,
                "needYield" : 0,
                "saveState" : 48,
                "restoreState" : 48,
                "isEOF" : 1,
                "invalidates" : 0,
                "docsExamined" : 6227,
```


Vediamo che l indice precedentemente inserito viene scelto dal sistema d' esecuzione.
Di conseguenza non avremmo più una scansione di tipo COOLSCAN, ma avremo un IXSCAN quindi una scansione attraverso il nostro indice rinominato `nazione_1`

notiamo che, oltre ad una riduzione del tempo impiegato abbiamo una logica riduzione dei dati esaminati. Passiamo da un tempo di 0.030 sec di mySql ad un tempo di 0.009 sec di mongoDB.

## #25 VOGLIAMO AVERE IL DATO RIGUARDANTE IL BIGLIETTO PIU' APPREZZATO DAI GIOVANI, COSI DA RIPROPORLO NELLA STAGIONE 2016

```
+---------+--------+
| Venduti | prezzo |
+---------+--------+
|    2342 |    150 |
|    2331 |     60 |
|    2320 |     90 |
|    2274 |    120 |
|    2224 |     30 |
|    2194 |    180 |
|    2170 |    210 |
+---------+--------+
7 rows in set (0,07 sec)
```

tempo query comprensivo di ottimizzazione.

## #25 MONGODB AGGREGATE

```
> db.spett.aggregate([
... {$match : { "eta" : { $lt : 22}}},
... {$group : {"_id" : "$biglietti.prezzo" , "venduti" : {$sum : 1}}},
... {$sort : {"venduti" : -1}}]).pretty()
```

```
{ "_id" : 90, "venduti" : 929 }
{ "_id" : 30, "venduti" : 917 }
{ "_id" : 210, "venduti" : 902 }
{ "_id" : 120, "venduti" : 899 }
{ "_id" : 60, "venduti" : 897 }
{ "_id" : 150, "venduti" : 894 }
{ "_id" : 180, "venduti" : 885 }
```

La differenza nella numerosità dei dati, è conseguenza della semplificazione apportata ( 1 spettatore ha 1 solo biglietto ) nel descrivere il ramo del db mySQL in mongoDB.

# JAVA

## GENERATORI

I dati presenti nei database, sono stati opportunamente costruiti tramite dei generatori in java. L' obbiettivo è quello di generare una gran quantità di dati mantenendo la consistenza, e le proprietà prefissate ad inizio progetto. Inoltre con le giusta combinazioni , abbiamo ottenuto dati "reali" in modo da facilitare la comprensione del database, e dei risultati derivanti dalle interrogazioni.

l' idea adottata per la creazione dei generatori, è schematizzata nei seguenti punti:

1. creazione file di testo, contenenti Nomi, Cognomi , Città , Nomi TV , Nomi sponsor.

2. per ogni file di testo creare un ArrayList di tipo adeguato (int,String)

3. Attraverso l uso di un Buffer, leggere da file ed inserire quanto letto negli ArrayList creati.

4. Creo un oggetto random per gestire la casualità dei dati, e creare varietà nel DB.

5. Grazie ad un oggetto PrintWriter e di un suo metodo .println( ) , creo una stampa multipla nella quale ogni riga rappresenterà una riga dell entità che si vuole rappresentare.

6. Inserire i dati nei database con le procedure apposite,
   ( INSERT , LOAD DATA , MONGOIMPORT ).

per quanto riguarda la creazione di file veritieri , complessi e di grandi dimensioni, non realizzabili manualmente, ad esempio, il file contenente 3000 nomi maschili, si è fatto uso di tool gratuiti online come Mockaroo ( www.mockaroo.com ) nella versione free.

## ESEMPIO CREAZIONE CAMPO COD_F  ED ENTITA' BIGLIETTO MYSQL

```
ArrayList<String> test = new ArrayList<String>(); // i cod_f sono 110000
```

```
/*---------- lettura per i codici fiscali ----------*/
    BufferedReader br = null;
    try {
        br = new BufferedReader(new FileReader("cf.txt"));
    } catch (FileNotFoundException e) {

        e.printStackTrace();
    }
    String s;

    try {
        while((s = br.readLine())!= null){
            //System.out.println(s);
             per = s.split(Pattern.quote(",")); // per splittare bene
             String s1 = per[0];
             //System.out.println(s1);
             test.add(s1);
            //System.out.println(per);
        }
    } catch (IOException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    }
    test.add(99999, "mntfnc12892abr");
    System.out.println(test.get(99999) + " ---> primo cod_f ");

/* --------- lettura per i codici fiscali finita ------- */
```

```
/*----------- SCRITTURA CAMPI Tab BIGLIETTO RESTANTI 150mila ----------*/
    String b1 = "B";
    PrintWriter scrivi1 = new PrintWriter(new File("biglietto2.txt"));
    //for (int j = 0; j < 100; j++) {
        for (int i = 100001; i <= 250000; i++) {
            scrivi1.println(
                 (i)+b1 + ","
                +  prezzi.get(r.nextInt((7-0)) + 0)  +  ","
                +  test.get(r.nextInt(100000 - 0) + 0) + ","
                + stadi.get(r.nextInt((357-0)) + 0) + ","
                );
        }

    scrivi1.close();
/* --------- FINE SCRITTURA BIGLIETTO ---------- */
```

* l' ArrayList con nome test è il contenitore dei cod_f .

# ESEMPIO CREAZIONE DOCUMENTS PER MONGODB

```
ArrayList<String> nomiM = new ArrayList<String>(); // i nomi maschili sono 3000
```

```
/*----------- lettura per i nomi maschili -----------*/

    try {
        br = new BufferedReader(new FileReader("nomiM.txt"));
    } catch (FileNotFoundException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    }

    try {
        while((s = br.readLine())!= null){
            //System.out.println(s);
             per = s.split("\n");
             nomiM.add(s);
            //System.out.println(per);
        }
    } catch (IOException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    }
    System.out.println(nomiM.get(0) + " ---> primo nomeM");

    br.close();

/*----------- fine lettura per i nomi maschili -----------*/
```

```
PrintWriter scrivi1 = new PrintWriter(new File("spett.json"));

/* test dato json */
for (int i = 0; i < 100000; i++) {

    String residenza = citys.get(r.nextInt((100 - 0) + 0));
    String citta = citys.get(r.nextInt((100- 0) + 0));

    scrivi1.println(
            "{" + "\"" + "_id" + "\"" + ":" + "\"" + test.get(i+1) + "\"" + ","+
            "\"" + "nome" + "\"" + ":" + "\"" + nomiF.get(r.nextInt((limSupNomi-limInfNomi)) + limInfNomi) + "\"" +","+
            "\"" + "cognome" + "\"" + ":" + "\"" + cogn.get(r.nextInt((limSupCogn-limInfCogn)) +limInfCogn) + "\""+","+
            "\"" + "sesso" + "\"" + ":" + "\"" +  f + "\"" + ","+
            "\"" + "eta" + "\"" + ":" +  (r.nextInt((etaSup-etaInf) + 1) + etaInf) + ","+
            "\"" + "nazione" + "\"" + ":" + "\"" + nations.get(r.nextInt((limSupNaz - limInfNaz)) + limInfNaz) + "\"" + ","+
            "\"" + "residenza" + "\"" + ":" + "\"" + residenza + "\"" + ","+
            "\"" + "biglietti" + "\"" + ":" + " {"+
            "\"" + "cod_b" + "\"" + ":" + "\"" +  (i+1)+"B" + "\"" +","+
            "\"" + "stadio" + "\"" + ":" + "{"+
            "\"" + "nome" + "\"" + ":" + "\"" + citta+ na.get(r.nextInt((3-0)+0))+ "\"" + ","+
            "\"" + "tetto" + "\"" + ":" + "\"" +bool.get(r.nextInt((2-0)) + 0) + "\"" + ","+
            "\"" + "superficie" + "\"" + ":" + "\"" + sup.get(r.nextInt((3-0)+0))+  "\"" + ","+
            "\"" + "citta" + "\"" + ":" + "\"" + citta + "\"" + "},"+
            "\"" + "prezzo" + "\"" + ":" +  prezzi.get(r.nextInt((7-0)) + 0) +  "}" + "}"
            );
}
scrivi1.close();
```