SQL

Corso base

Cosa è un database e un po' di storia

- A database is an organized collection of information
- A DBMS (DataBase Management System) is a program that stores, retrieves and modifies data in database on request
- RDBMS (relational)
  - Formalizzato da Codd nel 1970 nel paper "A relational model of data for large shared data banks"
  - https://it.wikipedia.org/wiki/12_regole_di_Codd
  - Collection of objects or relations that store the data
  - Set of operators that can act on the relations to produce other relations
  - Data integrity for accuracy and consistency

Prodotti in commercio

- https://en.wikipedia.org/wiki/Comparison_of_relational_database_management_systems
- IBM DB2 (System i Navigator)
- Microsoft Access (Office Access)
- Microsoft SQL Server (SQL Management Studio, TOAD for SQL Server)
- Oracle mySQL
- Oracle (Toad for Oracle, SQL Developer, sql plus)
- SAP HANA
- Teradata
- Netezza (Aginity Workbench for Netezza)

Tabella

- Definizione: A table is a collection of related data held in a structured format within a database
  - Struttura
  - Record
  - Campo (o colonna) e data type
  - Primary Key: una o più colonne che individuano univocamente un record. Ci possono essere più chiavi, ma solo una è principale

Esempi…

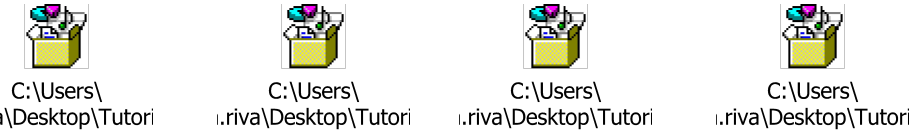
Structured Query Language (SQL)

SQL enables a programmer or database administrator to do the following:

- Create and modify a database's structure
- Change system security settings
- Add user permissions on databases or tables
- Query a database for information
- Update the contents of a database

Non è procedurale --> (PL/SQL program language)

Esempi…



select e from

Query: chi sono i dipendenti dell'azienda?

Select *Name* from *EMPLOYEE*

Select <nomi campi>

From <nome tabella>

select e from

- Select * --> tutti i campi di una tabella
- Select Occupation, Name, Age --> indicazione dei campi nell'ordine preferito
- Select Age as età from employee as dipend --> Alias
- Select Age*2 as doppio, Age/2 metà --> espressione
- Select distinct Age --> valori distinti, no doppi
- Select case when Age <= 40 then 'Giovane' else 'Vecchio' end
- Select decode(Occupation, 'Student', 1, 0) as flag

funzioni

- Aggregazione
  - Select count(*), max(Age), avg(Age) --> numero di record, valore max di Age, valore medio di Age
  - Select count(distinct Age) --> conta i valori distinti di Age
- Data e ora
  - Select add_month(now(),2) --> aggiunge 2 mesi alla data di oggi
  - Select last_day(add_month(now(),2)) --> ultimo giorno del mese
  - Sysdate, now() --> system day and hour (DB che usi funzione che trovi)

funzioni

- Aritmetiche
  - Select abs(a) --> valore assoulto
  - Select ceil(a), floor(a) --> intero più grande (piccolo) vicino
  - Select mod(a,b) --> resto della divisione a/b
  - Select sign(a) --> segno di a (-1 negativi, 1 positivi)
- Conversioni
  - Select to_char(5), to_char(sysdate,'yyyymmdd') --> converte numero (data) in stringa
  - Select to_number('5') --> converte in un numero
  - Select to_date('20151225','yyyymmdd') --> converte in formato data

funzioni

- Stringhe
  - Select concat(‘Blue ‘,’Bl’) --> concatena
  - Select upper(‘Blue’), lower(‘Bl’) --> converte in tutto maiuscolo, tutto minuscolo
  - Select lpad(‘Bl’,3,’ ‘), rpad(‘Blue’,10,’+’) --> aggiunge a sx (a dx) degli ‘ ‘ (‘+’) fino a 3 (10) caratteri totali
  - Select ltrim(‘ Bl’), rtrim(‘Blue+++’,’+’) --> taglia i caratteti ‘ ‘ a sx o l caratteti ‘+’ a dx
  - Select replace(‘aeiiou’,’i‘,’ ‘) --> sostituisce i con ‘ ‘ per ogni occorrenza
  - Select substr(‘aeiiou’,2,3) --> estrae una stringa dalla posizione 2 per 3 caratteri
  - Select instr(‘aeiiou’,’i') --> ritorna la posizione (numero) della stringa ‘i’
  - Select length(‘aeiiou’) --> ritorna la lunghezza della stringa

funzioni

Un pò di funzioni…

https://docs.oracle.com/cd/E17952_01/refman-5.0-en/func-op-summary-ref.html

https://docs.oracle.com/cd/B12037_01/server.101/b10759/functions001.htm

https://msdn.microsoft.com/it-it/library/ms177516(v=sql.120).aspx

where condition

Query: quanti sono gli ordini con più di 10 quantità ordinate?

Select count(*) from orders where Quantity > 10

Query: quali sono gli ordini con più di 10 quantità ordinate e ordinati di tipo BIKE SPEC oppure ordinati dopo il 1/6/1996?

Select * from orders

where Quantity > 10 and (Name = 'BIKE SPEC' or OrderedOn > '01-06-1996'

Select <nomi campi>

From <nome tabella>

Where <condizione>

where condition

Operatori:

- artmetici: +, -, *, /, MOD, ecc…
- confronto: =, >, >=, <, <=, in, like, between, exists
- operatori logici: AND, OR e NOT
- non è possibile usare funzioni di aggregazione direttamente nella where condition

where condition

- NULL value
  - Assenza di valore
  - Non è zero e non è blanck
  - Usare is null (is not null) per testare la presenza (assenza) del null
  - Null <operatore> <valore> è SEMPRE FALSA (es. Null >= 5)
  - Nvl: funzione per convertire un null (es. Nvl(Age,0))
- LIKE: cerca parte di una stringa
  - % stringa di lunghezza qualsiasi
  - _ stringa di un solo carattere

  Select * from EMPLOYEE where name like ‘%J%’

group by

Query: per ogni tipo di prodotto quanti pezzi mi hanno ordinato?

Select *Name, sum(Quantity) tot_qty*
from *ORDERS*
group by *Name*

Select <nomi campi>
From <nome tabella>
Group by <nomi campi>

group by

- Che differenza c'è?

```
Select Name from ORDERS group by Name

Select distinct Name from ORDERS
```

- Attenzione all'uso delle funzioni con la group by

```
Select substr(Name,1,3)
from ORDERS
group by substr(Name,1,3)
```

having

È la where condition sul risultato della group by

Select <nomi campi>

From <nome tabella>

Group by <nomi campi>

Having <condizione>

having

Query: quali sono i prodotti ordinati con più di 100 pezzi?

Select *Name, sum(Quantity) tot_qty*

from *ORDERS*

group by *Name*

Having sum(Quantity) > 100

order by

Query: chi sono i dipendenti dell'azienda ordinati per età?

Select *Name* from *EMPLOYEE* order by *Age*

Select <nomi campi>

From <nome tabella>

Order by <nomi campi>

order by

Esempi…

Select *Age, Name, Occupation*
from *EMPLOYEE*
order by *1* desc*, 2*

tutti i costrutti

Select <nomi campi> 4

From <nome tabella> 1

Where <condizione> 2

Group by <nomi campi> 3

Having <condizione> 4

Order by <nomi campi> 5

Join

Query: quanti pezzi sono stati consegnati all'indirizzo '10 OLDTOWN' ? (vedi tabelle CUSTOMER e ORDERS)

Select sum(quantity)
From *ORDERS* inner join *CUSTOMER* on *orders.name = customer.name*
Where *address* = '10 OLDTOWN'

Select <nomi campi>
From <nome tabella> **<tipo join>** <nome tabella> **on** <condizione di join>

Tipi di join

- INNER join: solo i record in comune (in base alle condizioni di join)
- LEFT OUTER JOIN: tabA left join tabB: tutti i record di tabA, i record di tabB sono valorizzati sono se le condizioni di join sono verificate
- RIGHT OUTER JOIN: tabA right join tabB: tutti i record di tabB, i record di tabA sono valorizzati sono se le condizioni di join sono verificate
- FULL OUTER JOIN: LEFT + RIGHT
- CROSS JOIN: ogni record di tabA è in relazione con ogni record di tabB (prodotto cartesiano)
- SELF-JOIN: join di una tabella con se stessa

Differenti sintassi

Standar ANSI

Select sum(quantity)
From *ORDERS* left join *CUSTOMER* on *orders.name = customer.name*
Where *address* = ‘10 OLDTOWN’

Sintassi Oracle

Select sum(quantity)
From *ORDERS , CUSTOMER*
Where *address* = ‘10 OLDTOWN’
and *orders.name = customer.name(+)*

Attenzione a dove sono i filtri

Che differenza c'è?

Select sum(quantity)
From *ORDERS* left join *CUSTOMER* on *orders.name = customer.name*
Where *address = '*10 OLDTOWN*'*

Select sum(quantity)
From *ORDERS* left join *CUSTOMER* on *orders.name = customer.name and address = '*10 OLDTOWN*'*

Select sum(quantity)
From *ORDERS* left join *(*select * from *CUSTOMER* where *address = '*10 OLDTOWN*')* on *orders.name = customer.name*

Subquery

```
Select sum(quantity)
From ORDERS
Where name = (Select max(name) From CUSTOMER Where address = '10 OLDTOWN')

Select sum(quantity)
From ORDERS
Where orders.name = (Select customer.name
                     From CUSTOMER
                     Where orders.name = customer.name and address = '10
OLDTOWN')

Select sum(quantity)
From ORDERS inner join
    (Select * From CUSTOMER Where address = '10 OLDTOWN') cust_oldtown
        on orders.name = cust_oldtown.name
```

Subquery e in/exists

```
SELECT DISTINCT store_name
FROM stores
WHERE stores.city in (SELECT city FROM cities_stores where stores.city_id =
cities_stores.id)

SELECT DISTINCT store_name
FROM stores
WHERE exists (SELECT 1 FROM cities_stores where stores.city_id = cities_stores.id)
```

Union, union all, minus e intersect

Select *Name* from *EMPLOYEE*

Union

Select *Name* from *FRIENDS*

- Union: unione dei risultati distinti
- Union all: unione dei risultati con gli eventuali valori doppi
- Minus: differenza dei risultati
- Intersect: risultati in comune

Esempi ed esercizi

- Pivot e reverse pivot
- Come individuare la chiave di una tabella
- Come verificare se una o più join sono corrette
- Come calcolare lo YTD

Pivot e reverse pivot

Pivot

```
SELECT Name,
        sum(case when type = 'Pos' then valore else 0 end) Pos,
        sum(case when type = 'Ab' then valore else 0 end) Ab,
        sum(case when type = 'Hits' then valore else 0 end) Hits,
        sum(case when type = 'Walks' then valore else 0 end) Ab
FROM TEAMSTATS_P
GROUP BY Name
```

Reverse pivot

```
SELECT Name, 'Pos' as type, Pos as valore
FROM TEAMSTATS
union all
SELECT Name, 'Ab' as type, Ab as valore
FROM TEAMSTATS
union all
SELECT Name, 'Hits' as type, Hits as valore
FROM TEAMSTATS
union all
SELECT Name, 'Walks' as type, Walks as valore
FROM TEAMSTATS;
```

Chiave tabella

```
SELECT Name, count(*)
from orders
group by Name
having count(*) > 1

SELECT Name, Orderedon, count(*)
from orders
group by Name, Orderedon
having count(*) > 1

SELECT Name, Orderedon, partnum, count(*)
from orders
group by Name, Orderedon, partnum
having count(*) > 1
```

Verifica correttezza delle join

```
select count(*)     --478642
from FACT_WW_spedizioni;

select count(*)     --478642   supponiamo invece 123456
from FACT_WW_spedizioni s   inner join DIM_TIPISPEDIZIONI tp  on s.fktiposped = tp.pktipispedizioni;
--andando in inner join ci perdiamo diversi record perchè non match
--questo tipo di join è potenzialmente sbagliata, cambiamo in outer join

select count(*)     --478642
from FACT_WW_spedizioni s  left join DIM_TIPISPEDIZIONI tp  on s.fktiposped = tp.pktipispedizioni;
--ritorniamo al numero corretto di record
--per vedere quali spedizioni non hanno il corrispondente record sulla DIM aggiungiamo una where condition

select s.*
from FACT_WW_spedizioni s  left join DIM_TIPISPEDIZIONI tp  on s.fktiposped = tp.pktipispedizioni
where tp.pktipispedizioni is null;


--supponiamo ora che la left ritorni un numero di record più alto per esempio 478650, 8 record in più
--vuol dire che c'è un potenziale problema sulle condizioni di join, per un record della FACT ci sono due o più record sulla DI
--per individuare questi record si raggruppa sul/sui campi chiave della FACT e si contano i record

select NumSped, TipoSped, DataSped, BdA, count(*)
from FACT_WW_spedizioni s  left join DIM_TIPISPEDIZIONI tp  on s.fktiposped = tp.pktipispedizioni
group by NumSped, TipoSped, DataSped, BdA
having count(*) > 1;
--individuati i casi di "doppio" li si filtrano per evidenziare le anomalie o correzioni da apportare ai campi di join
```

Come calcolare lo YTD

```
select sum(W3ORIMPL) lordo, sum(W3ORIMPN) netto, count(*)
from LUXDWWRK..FD3ORDINE_OK_OAKLEY
where W3ORFILI = '000008'
and W3ORWHRE = 'US30'
and W3ORSEMT in ('OO','OX');

select DTTMM, sum(W3ORIMPL) lordo, sum(W3ORIMPN) netto, count(*)
from LUXDWWRK..FD3ORDINE_OK_OAKLEY
     INNER JOIN LUXDWDAT..DMDTT ON (FD3ORDINE_OK_OAKLEY.SKDCRE7ID=LUXDWDAT.DMDTT.DWID  AND  LUXDWDAT.DMDTT.DTTLVL='7')
where W3ORFILI = '000008'
and W3ORWHRE = 'US30'
and W3ORSEMT in ('OO','OX')
and DTTYY = 2014
group by DTTMM
order by 1;

select c.DTTMM, sum(W3ORIMPL) lordo, sum(W3ORIMPN) netto, count(*)
from LUXDWWRK..FD3ORDINE_OK_OAKLEY
     INNER JOIN LUXDWDAT..DMDTT a ON (FD3ORDINE_OK_OAKLEY.SKDCRE7ID=a.DWID  AND  a.DTTLVL='7')
     INNER JOIN (select distinct DTTMM from LUXDWDAT..DMDTT where DTTYY = 2014 and DTTLVL='7') c ON (a.DTTMM<=c.DTTMM)
where W3ORFILI = '000008'
and W3ORWHRE = 'US30'
and W3ORSEMT in ('OO','OX')
and a.DTTYY = 2014
group by c.DTTMM
order by 1;
```

Structured Query Language (SQL)

SQL enables a programmer or database administrator to do the following:

- Create and modify a database's structure
- Change system security settings
- Add user permissions on databases or tables
- Query a database for information
- Update the contents of a database

Inserire nuovi record

- Inserire un singolo record

Insert into *EMPLOYEE (Name)* values (‘Icardi’)

Insert into <nome tabella> (<nomi campi>)
Values (<dati>)

Attenzione vanno rispettate tutte le constraint

Inserire nuovi record

- Inserire uno o più record da un'tabella/query

Insert into *EMPLOYEE_NEW*
Select * from *EMPLOYEE*

Insert into <nome tabella> (<nomi campi>)
Select <nomi campi>
From <nome tabella>

Attenzione rispettare l'ordine dei campi tra la insert e la select

Modifica record

Update *EMPLOYEE Set* age = age+10

Update *EMPLOYEE Set* age = 23, Occupation = ‘Player’ where Name = ‘Icardi’

Update <nome tabella>

Set <nomi campi> = espressione

Modifica record

con valori presi da altre tabelle

(sql server)

```
UPDATE SalesPerson
SET SalesYTD = SalesYTD + SubTotal
FROM SalesPerson AS sp
JOIN SalesOrderHeader AS so
   ON sp.BusinessEntityID = so.SalesPersonID;
```

(oracle)

```
UPDATE SalesPerson sp
SET SalesYTD = SalesYTD + (SELECT SubTotal
                           FROM SalesOrderHeader AS so
                           WHERE sp.BusinessEntityID = so.SalesPersonID)
WHERE exists (SELECT 1
              FROM SalesOrderHeader AS so
              WHERE sp.BusinessEntityID = so.SalesPersonID);
```

Copia di tabelle

Per esempio per fare un backup

(sql server)
Select *
Into *EMPLOYEE_NEW*
from *EMPLOYEE*

(oracle)
Create table *EMPLOYEE_NEW* as
Select * from *EMPLOYEE*