SQL

Structured Query Language

Prof. Claudio Capiluppi - Facoltà di Scienze della Formazione - A.A. 2009/10

SQL

SELECT fields FROM table WHERE condizione		seleziona i campi (colonne) da visualizzare specifica la tabella da cui leggere i dati seleziona i record (righe) da visualizzare
SELECT expression FROM table		è possibile specificare espressioni, contenenti operazioni e funzioni di calcolo (campi calcolati)
SELECT [field name] FROM [table name]		se il nome di un campo o di una tabella contiene spazi o altri caratteri speciali o coincide con una KEYWORD del linguaggio SQL: deve essere indicato tra parentesi [ ]
SELECT table.field FROM tables		occorre specificare la tabella di origine del campo quando i dati provengono da più tabelle

SQL

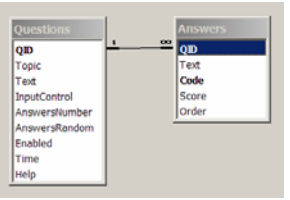
- WHERE
  - WHERE Code = 2
  - WHERE QID = 'Q0004' AND [Date] > #12/1/2005#
  - WHERE QID = 'Q0004' OR QID = 'Q0013'
  - WHERE Active = True
  - WHERE QID IN ("Q0003", "Q0004", "Q0011", "Q0044")
  - WHERE QID LIKE "Q000?"
  - WHERE QID LIKE "Q0*"
- LIKE: metacaratteri (*wildcard*)
  - * zero o più caratteri qualsiasi (* in MS Access, % in SQL standard)
  - ? un singolo carattere qualsiasi (_ in SQL standard)
  - # una singola cifra (0-9)
  - [*charlist*] qualsiasi singolo carattere in *charlist*
  - [!*charlist*] qualsiasi singolo carattere NON in *charlist*

Prof. Claudio Capiluppi - Facoltà di Scienze della Formazione - A.A. 2009/10

SQL

- PARAMETRIZZAZIONE DI UNA QUERY
- Per generalizzare una query (ad es. una ricerca) è possibile utilizzare come operando in una espressione, al posto di un valore fisso, un PARAMETRO variabile.
- Esempi:
  - WHERE QID = pippo
  - WHERE QID = ???
  - WHERE QID = [Inserisci il QID da cercare:]
  - WHERE [Text] LIKE [Stringa di ricerca (con metacaratteri):]
- Qualsiasi nome non riconosciuto dall'interprete SQL come nome di campo (o come keyword del linguaggio stesso) viene interpretato come un parametro: il valore del parametro viene richiesto interattivamente quando la query viene mandata in esecuzione.

JOIN



- JOIN INTERNO (INNER):

```
SELECT    Questions.QID, Questions.Text, Answers.Text
FROM      Questions  INNER JOIN  Answers
          ON  Questions.QID = Answers.QID
```

- JOIN ESTERNO (LEFT/RIGHT):

```
SELECT    Questions.QID, Questions.Text, Answers.Text
FROM      Questions  LEFT JOIN  Answers
          ON  Questions.QID = Answers.QID
```

- EQUI-JOIN = PRODOTTO CARTESIANO + SELEZIONE:

```
SELECT    Questions.QID, Questions.Text, Answers.Text
FROM      Questions, Answers
WHERE     Questions.QID = Answers.QID
```


UNION

- SELECT field(s) FROM table1
  UNION
  SELECT field(s) FROM table2
- La UNION è un altro tipo di operazione di unione tra tabelle rispetto al JOIN: in questo caso le righe della seconda tabella vengono accodate a quelle della prima, colonna per colonna
  - l'operazione ha senso quando le colonne corrispondenti delle due tabelle rappresentano gli stessi attributi
  - il risultato è l'insieme dei record (righe) delle due tabelle
- Si usa tipicamente per riunire i record di due o più tabelle che hanno esattamente la stessa struttura (cioè gli stessi attributi):

```
SELECT * FROM anagVR
UNION
SELECT * FROM anagVI
UNION
SELECT * FROM anagPD
```

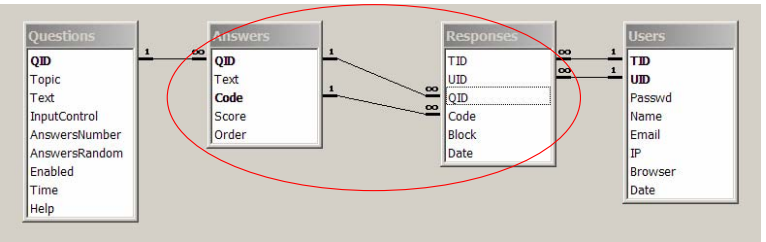
SQL

- INNER JOIN:

```
SELECT   table1.fields, table2.fields
FROM     table1 INNER JOIN table2 ON table1.field=table2.field

SELECT   table1.fields, table2.fields
FROM     table1 INNER JOIN table2
         ON  table1.field1=table2.field1  AND  table1.field2=table2.field2
```

- Esempio:




```
SELECT   Responses.QID, Responses.Code, Answers.Score
FROM     Answers INNER JOIN Responses
         ON  Answers.QID=Responses.QID AND Answers.Code=Responses.Code
```

Prof. Claudio Capiluppi - Facoltà di Scienze della Formazione - A.A. 2009/10

SQL

- JOIN SEMPLICE: tra due tabelle

```
SELECT   table1.fields, table2.fields
FROM     table1 INNER JOIN table2 ON table1.field=table2.field
```

- JOIN DOPPIO: tra tre tabelle

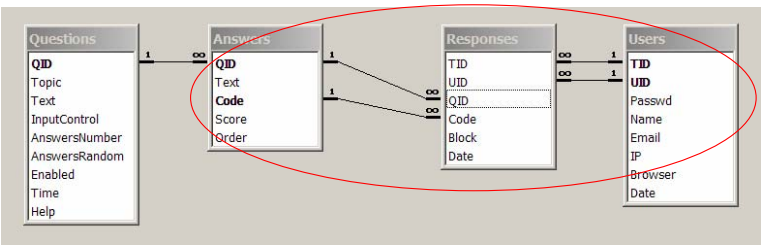
```
SELECT   table1.fields, table2.fields, table3.fields
FROM     (table1 INNER JOIN table2 ON table1.field=table2.field)
         INNER JOIN table3 ON table2.field=table3.field
```

oppure anche:

```
SELECT   table1.fields, table2.fields, table3.fields
FROM     table1 INNER JOIN
         (table2 INNER JOIN table3 ON table2.field=table3.field)
         ON table1.field=table2.field
```

SQL

- Esempio: Join doppio tra Answers, Responses e Users




```
SELECT  Users.Name, Responses.QID, Responses.Code, Answers.Score
FROM    (Answers INNER JOIN Responses ON Answers.QID=Responses.QID
        AND Answers.Code=Responses.Code) INNER JOIN Users
        ON Responses.UID=Users.UID AND Responses.TID=Users.TID
```

oppure:

```
FROM    Answers INNER JOIN (Responses INNER JOIN Users ON
        Responses.UID=Users.UID AND Responses.TID=Users.TID)
        ON Answers.QID=Responses.QID AND Answers.Code=Responses.Code
```

Prof. Claudio Capiluppi - Facoltà di Scienze della Formazione - A.A. 2009/10

SQL

- QUERY CON ORDINAMENTO:

```
SELECT   fields
FROM     tables
ORDER BY fields  DESC
```

- ```
  SELECT   Questions.QID, Questions.Text, Answers.Text, Answers.Score
  FROM     Questions INNER JOIN Answers ON Questions.QID=Answers.QID
  ORDER BY Questions.QID
  ```

- ```
  SELECT   Questions.QID, Questions.Text, Answers.Text, Answers.Score
  FROM     Questions INNER JOIN Answers ON Questions.QID=Answers.QID
  ORDER BY Questions.QID DESC, Answers.Code
  ```

SQL

- QUERY DI AGGREGAZIONE:

```
SELECT    fields, funzione_di_aggregazione(field)
FROM      tables
GROUP BY  fields
```

- FUNZIONI DI AGGREGAZIONE:

  COUNT(), SUM(), AVG(), STDEVP(), MAX(), MIN(), FIRST(), LAST(), …

- Esempi

-
```
SELECT    Questions.QID, COUNT(Answers.QID) AS NR
FROM      Questions INNER JOIN Answers ON Questions.QID=Answers.QID
GROUP BY  Questions.QID
```

-
```
SELECT    Questions.QID, FIRST(Questions.Text) AS T, COUNT(*) AS N
FROM      Questions INNER JOIN Answers ON Questions.QID=Answers.QID
GROUP BY  Questions.QID
```

Prof. Claudio Capiluppi - Facoltà di Scienze della Formazione - A.A. 2009/10

SQL

- QUERY DI AGGREGAZIONE:

```
SELECT    fields, funzione_aggregazione(field)
FROM      tables
WHERE     condizione
GROUP BY  fields
HAVING    condizione
ORDER BY  field
```

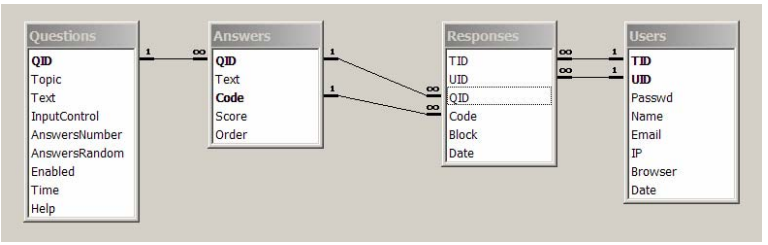
- Esempio: Ricerca libri disponibili in biblioteca

```
SELECT    Libri.ID, FIRST(Libri.Titolo), LAST(Prestiti.DataPrestito),
          LAST(Prestiti.DataRestituzione)
FROM      Libri LEFT JOIN Prestiti ON Libri.ID=Prestiti.IDLibro
WHERE     Libri.Titolo LIKE [Titolo libro da cercare]
GROUP BY  Libri.ID
HAVING    LAST(Prestiti.DataRestituzione) IS NOT NULL
ORDER BY  FIRST(Libri.Titolo)
```

SQL

- Esempio: Correzione dei compiti

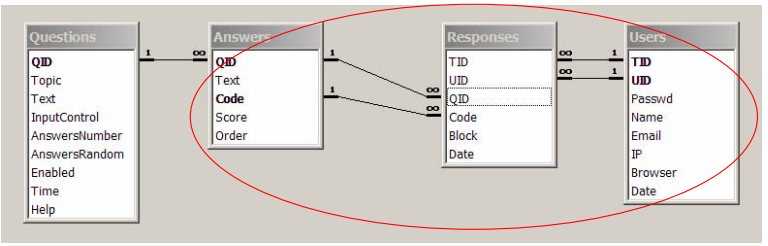
```
SELECT   UID, COUNT(Responses.QID) AS N, SUM(Answers.Score) AS V
FROM     Responses INNER JOIN Answers
         ON Responses.QID=Answers.QID AND Responses.Code=Answers.Code
WHERE    Responses.TID = "2008-01-08"
GROUP BY UID
```



SQL

- Esempio: Correzione dei compiti (aggiungiamo i nomi degli studenti)

```
SELECT   Users.UID, FIRST(Users.Name) AS Nome,
         COUNT(Responses.QID) AS N, SUM(Answers.Score) AS V
FROM     (Responses INNER JOIN Answers
         ON Responses.QID=Answers.QID AND Responses.Code=Answers.Code)
         INNER JOIN Users
         ON Users.UID=Responses.UID AND Users.TID=Responses.TID
WHERE    Users.TID="2008-01-08"
GROUP BY Users.UID
ORDER BY FIRST(Users.Name)
```



SQL

- QUERY NIDIFICATE
- SELECT fields
  FROM (SELECT fields FROM table …)
- SELECT table.fields, query.fields
  FROM table INNER JOIN (SELECT …) AS query
  ON table.field = query.field
- SELECT fields
  FROM table
  WHERE field IN (SELECT …)

SQL

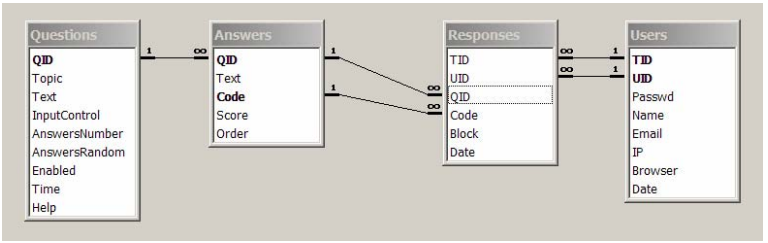
- Esempio: Ricerca libri disponibili in biblioteca

```
SELECT   Libri.ID, Libri.Titolo
FROM     Libri
WHERE    Libri.Titolo LIKE [Titolo libro da cercare]
         AND Libri.ID NOT IN (SELECT IDLibro FROM Prestiti
                              WHERE DataRestituzione IS NULL)
```

SQL

- Esempio: Correzione dei compiti (soluzione alternativa)

```
SELECT    Users.Name, Results.*
FROM      Users  LEFT JOIN
(SELECT   UID, TID, COUNT(Responses.QID) AS N, SUM(Answers.Score) AS V
FROM      Responses INNER JOIN Answers
ON        Responses.QID=Answers.QID AND Responses.Code=Answers.Code
GROUP BY  Responses.UID, Responses.TID)  AS  Results
ON        Results.UID=Users.UID AND Results.TID=Users.TID
ORDER BY  Users.Name
```



SQL

- CROSSTAB QUERY: query di riepilogo a campi incrociati

```
TRANSFORM  funzione_aggregazione(field)
SELECT     fields
FROM       tables
GROUP BY   fields
PIVOT      field
```

-
```
TRANSFORM  Count(Code)
SELECT     UID, Count(Code)
FROM       Responses
GROUP BY   UID
ORDER BY   UID
PIVOT      TID
```

- PIVOT Year([Date])
- PIVOT Format([Date],"yymm")

SQL

- La documentazione migliore si trova on-line (in inglese)

- Microsoft Jet SQL
  - Office:
    http://office.microsoft.com/en-us/assistance/CH062526881033.aspx

- Molti altri siti, es:
  - Devguru:
    http://www.devguru.com/Technologies/jetsql/quickref/jet_sql_list.html
  - Wikipedia (inglese):
    http://en.wikipedia.org/wiki/Sql
  - Wikipedia (italiana):
    http://it.wikipedia.org/wiki/Sql