Interrogazioni

1. Selezione

`SELECT attributi FROM tabelle [WHERE condizione]`

WHERE
Condizione: Espressioni booleane semplici mediante operatori logici AND, OR, NOT.

Altri operatori possibili
- `[NOT] BETWEEN a AND b`
- `[NOT] IN`
- `IS [NOT] NULL`
- `[NOT] LIKE 'pattern'`
- `[NOT] EXISTS`

Pattern Matching
- `_` rappresenta un carettere arbitrario
- `%` rappresenta una stringa di lunghezza arbitraria

Ridenominazione
`SELECT nome, cognome, stipendio AS nuovoStipendio`

Eliminazione dei duplicati
`SELECT DISTINCT`

Ordinamento del risultato
`ORDER BY attributo ASC|DESC`

Oss. Default è ASC

2. JOIN

```
SELECT Impiegato .nome, Impiegato.cognome, Dipartimento.indirizzo FROM Impiegato, Dipartimer
```

Oss. Questo equivale ad effettuare il prodotto cartesiano delle tabelle specificate nel campo FROM e ad applicare la condizione esplicitata in WHERE.

Oss. Nel campo WHERE possono essere specificate sia le condizioni di JOIN che le condizioni di selezione.

```
SELECT Impiegato.nome, Impiegato.cognome, Dipartimento.indirizzo FROM Impiegato, Dipartiment
```

Utilizzo di ALIAS
Ridenominazione standard:

```
SELECT I.nome, I.cognome, Indirizzo FROM Impiegato AS I, Dipartimento AS D WHERE I.dipN = D
```

Uso Avanzato per confrontare campi di tabelle con se stessi:

```
SELECT I.nome, I.cognome, I.ufficio FROM Impiegato AS I, Impiegato AS Smith WHERE I.ufficio
```

Tipologie

- **(INNER) JOIN** - coincide con Theta-join
- **NATURAL JOIN**

- **RIGHT, LEFT, FULL OUTER JOIN**

INNER JOIN
`FROM Impiegato INNER JOIN Dipartimento ON Impiegato.dipN = Dipartimento.nome`

Questo coincide con
`SELECT * FROM Impiegato, Dipartimento WHERE Impiegato.dipN = Dipartimento.nome`

NATURAL JOIN
Nel NATURAL JOIN non viene specificata nessuna condizione di join. Viene applicata una condizione implicita di equijoin per ciascuna coppia di attributi con lo stesso nome nelle due relazioni.

Oss. Ogni tupla viene riportata una sola volta nel risultato. I campi con nomi diversi possono essere comunque utilizzati nel join rinominandoli.

OUTER JOINS
Esegue un join mantenendo nel risultato tutte le righe che fanno parte di una o di entrambe le relazioni, estendendo gli attributi che non trovano riscontro con valori NULL.

LEFT (OUTER) JOIN
Fornisce il risultato del join esteso con le tuple della relazione che compare a sinistra del JOIN

RIGHT (OUTER) JOIN
Fornisce il risultato del join esteso con le tuple della relazione che compare a destra del JOIN

FULL (OUTER) JOIN
Fornisce il risultato del join esteso con le tuble di entrambe le relazioni.

Es.
`SELECT D.nome FROM Dipartimento AS D LEFT JOIN Impiegato AS I ON D.direttore = I.id`

3. Operatori Aggregati

SQL mette a disposizione una serie di operatori aggregati per calcolare valori aggregati da insiemi di tuble di relazioni.

Operatori:
- **Count**: cardinalità
- **SUM**: sommatoria
- **MAX**: massimo
- **MIN**: minimo
- **AVG**: media

es. `SELECT COUNT * FROM Impiegato WHERE dipN = 'Prod'` Restituisce il numero di dipendenti nel dip di produzione

Raggruppamenti
`GROUP BY lista attributi`

Necessità di applicare operatori aggregati e sottogruppi di tuple di una relazione in base al valore di uno o più attributi. Prima viene effettuato il raggruppamento, poi viene applicato l'operatori aggregato a ciascun sottogruppo.

es.
`dipN, COUNT(*) as TotImpiegati, SUM(stipendio) AS TotStipendio FROM Impiegato GROUP BY dipN`

Having
Specifica una condizione su un gruppo di tuple associata al valore degli attributi di raggruppamento.

es.
`SELECT dipN, SUM(stipendio) FROM Impiegato GROUP BY dipN HAVING SUM(stipendio)>130`
oss. `WHERE` indica una condizione sulle tuple, `HAVING` invece una condizione sui raggruppamenti.

4. Interrogazioni Insiemistiche

Le interrogazioni insiemistiche UNION, INTERSECT, EXCEPT sono incorporate in SQL. Di *default* i duplicati vengono eliminati. Si può fornire il parametro *ALL* per includere i duplicati nelle tuple restituite.

UNION
`SELECT nome FROM Impiegato UNION SELECT cognome FROM Impiegato`, senza duplicati

`SELECT nome FROM Impiegato UNION ALL SELECT cognome FROM Impiegato`, include i duplicati

5. Interrogazioni Nidificate

Argomento di `WHERE` è un valore confrontato con il risultato di una query eseguita internamente.

es.
```
SELECT Impiegato.nome, Impiegato.cognome FROM Impiegato, Dipartimento WHERE Impiegato.dipN =
```

Questa query può essere espressa in **forma nidificata**:
```
SELECT nome, cognome FROM Impiegato WHERE dipN IN (SELECT nome FROM Dipartimento WHERE indi
```

6. Subquery Correlate

Subquery utilizzate in WHERE.

es.
```
SELECT DISTINCT nome, cognome, stupendio, dipN FROM Impiegato AS X WHERE Stipendio > (SELEC
```

oss. La subquery correlata non è una subquery standard. Viene infatti eseguita più volte. Ogni volta che la query esterna trova una tupla candidata nella query esterna, chiama la query interna che correla uno o più dei propri parametri con gli attributi della query esterna.

EXISTS
Restituisce valore booleano true/false se la subquery restituisce almeno una tupla.

```
SELECT ID, cognome, nome FROM Impiegato I1 WHERE EXISTS (SELECT * FROM Impiegato I2 WHERE I
```

Manipolazione

1. Inserimento
`INSERT INTO nomeTabella [attributi] VALUES (valori)`

2. Modifica
```
UPDATE nomTabella SET attributo = < espressione | SSELECTSQL | NULL > {, attributo...} [WHE
```

3. Cancellazione

`DELETE FROM nomeTabella WHERE attributo='param'` **oss.** Senza condizione, vengono cancellate tutte le tuple.