

SQL

- DML
- TCL
- DDL
- Indici
- View
- Progetto di riferimento
 - <https://github.com/egalli64/jd> – folder mySql (*modulo 4*)

INSERT

INSERT INTO table_name (columns...) **VALUES** (values...);

insert into clients (client_id, name, nickname) values (20, 'Mordor Shifty Solutions', 'Moh');

- È possibile inserire più righe con un solo statement

insert into clients (client_id, name, nickname) values

(21, 'East Gondor Real Estate', 'Gondy'),

(22, 'Rohan Horse Equipments', 'Ron');

- I valori con default (NULL o altro), sono impliciti

insert into clients (client_id, name) values (12, 'Kerr & Reetch Associates');

insert into clients (name) values ('Multiple Oz Factories');

- Il nome delle colonne è opzionale (cfr. DESCRIBE)

insert into clients values (13, 'Rainydays Tour Operator', null);

UPDATE (WHERE!)

UPDATE table_name

SET column = value [, column2 = value2 ...]

[**WHERE** condition];

```
update clients
```

```
set nickname = concat('Client ', client_id)
```

```
where client_id > 10;
```

- Nella clausola WHERE, quando possibile, conviene operare sulla PK
 - Meglio se la selezione avviene via uguaglianza, selezionando così una singola riga

MySQL Workbench check su UPDATE/DELETE

Edit → Preferences... → SQL Editor → (in fondo al tab) Safe Updates

DELETE (WHERE!)

DELETE FROM table_name

[**WHERE** condition];

```
delete from clients  
where client_id > 10;
```

- Nella clausola WHERE, quando possibile, conviene operare sulla PK
 - Meglio se la selezione avviene via uguaglianza, selezionando così una singola riga

MySQL Workbench check su UPDATE/DELETE

Edit → Preferences... → SQL Editor → (in fondo al tab) Safe Updates

Transazioni

- Vogliamo eseguire più comandi **DML** in un unico blocco **ACID**
 - **A**tomic **C**onsistent **I**solated **D**urable
- MySQL default autocommit: ogni DML è in una propria transazione
 - **SET AUTOCOMMIT** per specificare il comportamento nella sessione corrente (0 / 1)
 - **START TRANSACTION** disabilita temporaneamente l'autocommit fino al termine della transazione
 - MySQL Workbench Query → Auto-Commit Transactions
 - Eclipse Database Development: Window, Preferences, Data Management, SQL Development, SQL Editor, SQL Files / Scrapbooks, Connection Commit Mode → Manual
- Se autocommit è 0 (false) anche **BEGIN** causa l'inizio di una nuova transazione
- Una transazione termina esplicitamente con **COMMIT**, **ROLLBACK**
 - implicitamente alla prima istruzione DDL, DCL, EXIT (→ COMMIT o ROLLBACK in caso di failure)

COMMIT, ROLLBACK, SAVEPOINT

SAVEPOINT: punto intermedio in una transazione

```
insert into clients (name) values ('Kerr & Reetch Associates');  
savepoint sp;
```

```
insert into clients (name) values ('Oz Singleton Factories');
```

```
rollback to sp; -- keep K&R, rollback Oz
```

```
commit; -- persist K&R
```

Livelli di isolamento nelle transazioni

- Transazioni concorrenti possono causare problemi in lettura:
 - **Phantom read**: T1 SELECT su più righe; T2 INSERT o DELETE nello stesso intervallo; T1 riesegue la stessa SELECT, nota un fantasma (apparso o scomparso) nel risultato
 - **Non repeatable read**: T1 SELECT, T2 UPDATE, T1 SELECT non ripetibile
 - **Lost update**: T1 e T2 SELECT, T1 UPDATE, T2 UPDATE. Il primo update è perso
 - **Dirty read**: T1 UPDATE, T2 SELECT, T1 ROLLBACK, valore per T2 è invalido
- Garanzie fornite da DBMS
 - READ UNCOMMITTED**: tutti comportamenti leciti
 - READ COMMITTED**: impedisce solo dirty read
 - REPEATABLE READ**: phantom read permesse ← default MySQL
 - SERIALIZABLE**: nessuno dei problemi indicati ← default SQL
- Impostazione del comportamento per sessione o globale: SET TRANSACTION ISOLATION LEVEL
- Lock su SELECT: FOR SHARE / FOR UPDATE + NOWAIT / SKIP LOCKED

CREATE TABLE

- Ricordarsi di operare sullo schema/database corretto
 - Ex: use me;
- Nome tabella, nome e tipo colonne, constraint, ...
create table items (
 item_id integer primary key,
 status char,
 name varchar(20),
 exec_id integer);

CREATE TABLE AS SELECT

- Se si hanno i privilegi in lettura su una tabella si possono copiare dati e tipo di ogni colonna

(GRANT SELECT ON ... TO ...)

create table execs

as

```
select employee_id as exec_id, first_name, last_name, hire_date, salary  
from employees  
where department_id = 90;
```

ALTER TABLE

- ADD / DROP COLUMN

`alter table items add counter decimal(38, 0);`

`alter table items drop column counter;`

- ADD CONSTRAINT CHECK / UNIQUE

`alter table items add constraint items_status_ck check(status in ('A', 'B', 'X'));`

- `alter table execs add constraint execs_name_uq unique(first_name, last_name);`

- ADD CONSTRAINT PRIMARY KEY / senza o con AUTO_INCREMENT

- `alter table execs add constraint primary key(exec_id);`

- `alter table execs modify exec_id int primary key auto_increment;`

CREATE TABLE con CONSTRAINT

```
create table details (  
  detail_id integer primary key  
    constraint detail_id_ck check (mod(detail_id, 2) = 1),  
  status char default 'A'  
    constraint detail_status_ck check (status in ('A', 'B', 'X')),  
  -- alternativa: status enum ('A', 'B', 'X') default 'A'  
  name varchar(20),  
    -- not null,  
    -- unique,  
  coder_id integer,
```

```
  constraint details_exec_fk foreign key (exec_id) references execs (exec_id), -- on delete cascade / set null
```

```
  constraint details_name_status_uq unique (name, status)
```

```
);
```

Attributi per colonne:

NOT NULL
UNIQUE
DEFAULT
AUTO_INCREMENT
...

TRUNCATE / DROP TABLE

MySQL Workbench ha “safe mode” che limita le funzionalità standard
(Edit → Preferences → SQL Editor → Safe Updates)

- `delete from` table_name; -- DML → rollback
- `truncate table` table_name; -- no rollback!
- `drop table` table_name; -- no rollback!
- Negli script che si pensa possano essere eseguiti più volte è spesso utile fare un check sull'esistenza della tabella prima della sua eliminazione
 - `drop table if exists` table_name;

INDEX

- Possono velocizzare l'accesso alle tabelle, riducendo gli accessi alla memoria di massa

- **B-Tree** by default

- indice semplice

- create index** execs_last_name_ix **on** execs(last_name);

- indice composto

- create index** execs_name_ix **on** execs(first_name, last_name);

- drop index** execs_last_name_ix **on** execs;

VIEW

- Query predefinita su una o più tabelle
 - acceduta come se fosse una tabella standard
- Semplifica e controlla l'accesso ai dati

```
create [or replace] view junior_execs_view as  
    select first_name, last_name, hire_date from execs  
    where exec_id != 100;
```

```
drop view odd_execs_view;
```

Esercizi

- Coders
 - Inserire come assunti oggi:
 - 201, Maria Rossi, 5000€ e 202, Franco Bianchi, 4500€
 - Cambiare il nome da Maria a Mariangela
 - Aumentare di 500€ i salari minori di 6000€
 - Eliminare Franco Bianchi
 - Committare i cambiamenti