# SQL

- DBMS
  - Database relazionali: MySql
  - Relazioni tra tabelle
- SQL
  - Amministrazione di database
  - Tipi di dato
  - Comandi (DML, DDL, …)
    - SELECT
  - Operatori
- Progetto di riferimento
  - https://github.com/egalli64/mpjp mySql *(modulo 1)*

# Database Management System

- Principali DBMS Relazionali
  - Oracle, MySQL, SQL Server, PostgreSQL, DB2
- NoSQL
  - MongoDB (doc), ElasticSearch (doc), Redis (k-v)

# MySQL

https://www.mysql.com/downloads/
https://dev.mysql.com/downloads/
https://dev.mysql.com/downloads/installer/

https://dev.mysql.com/doc/

# Alcuni IDE per MySQL

- Quest Toad Edge
- MySQL Workbench
- Database Development per Eclipse
  - Help, Install New Software, Work with (...) → Database Development
- DBeaver (standalone o plugin per Eclipse)
- Accesso CLI (mysql.exe nella directory MySQL server bin)

  mysql -u root -p

  "C:\Program Files\MySQL\MySQL Server 8.0\bin\mysql" -u root -p

# Database Relazionale

- Colonna: un singolo tipo di dato (campo) memorizzato in una tabella
- Riga (o record): collezione di dati (colonne) che descrivono completamente un'entità
- Tabella: insieme di righe in memoria volatile (result set) o persistente
- Tabelle memorizzate in uno schema del database, associato ad un utente
- Relazioni tra tabelle: primary key (PK) → foreign key (FK)
- PK: identifica univocamente (naturale o surrogata) una riga nella tabella corrente (normalmente singola colonna)
- FK: identifica univocamente una riga in un'altra tabella
- Un utente può avere il permesso di accedere tabelle di altri schemi
- SQL è il linguaggio standard (con variazioni) per l'accesso a database relazionali

# Relazioni tra tabelle

- **One to many** / **many to one**
  - Uno stato (PK) → molte città (FK duplicata)
- **Many to many** (implementato via tabella intermedia)
  - Uno stato → molte organizzazioni
  - Una organizzazione → molti stati
- **One to one**
  - Uno stato (PK) → una capitale (FK unique)

È compito del DBMS mantenere l'integrità referenziale

# SQL

- DQL – Data Query Language
  - SELECT
- DML – Data Manipulation Language
  - INSERT, UPDATE, DELETE
- DDL – Data Definition Language
  - CREATE, ALTER, DROP, RENAME, TRUNCATE
- TC – Transaction Control
  - COMMIT, ROLLBACK, SAVEPOINT
- DCL – Data Control Language
  - GRANT, REVOKE

Le keyword SQL sono *case insensitive*

select = SELECT

# Amministrazione DBMS

```
drop user if exists me;
drop schema if exists me;

create user me identified by 'password';
create schema me;
grant all privileges on me.* to me;
grant alter routine on me.* to me;
```

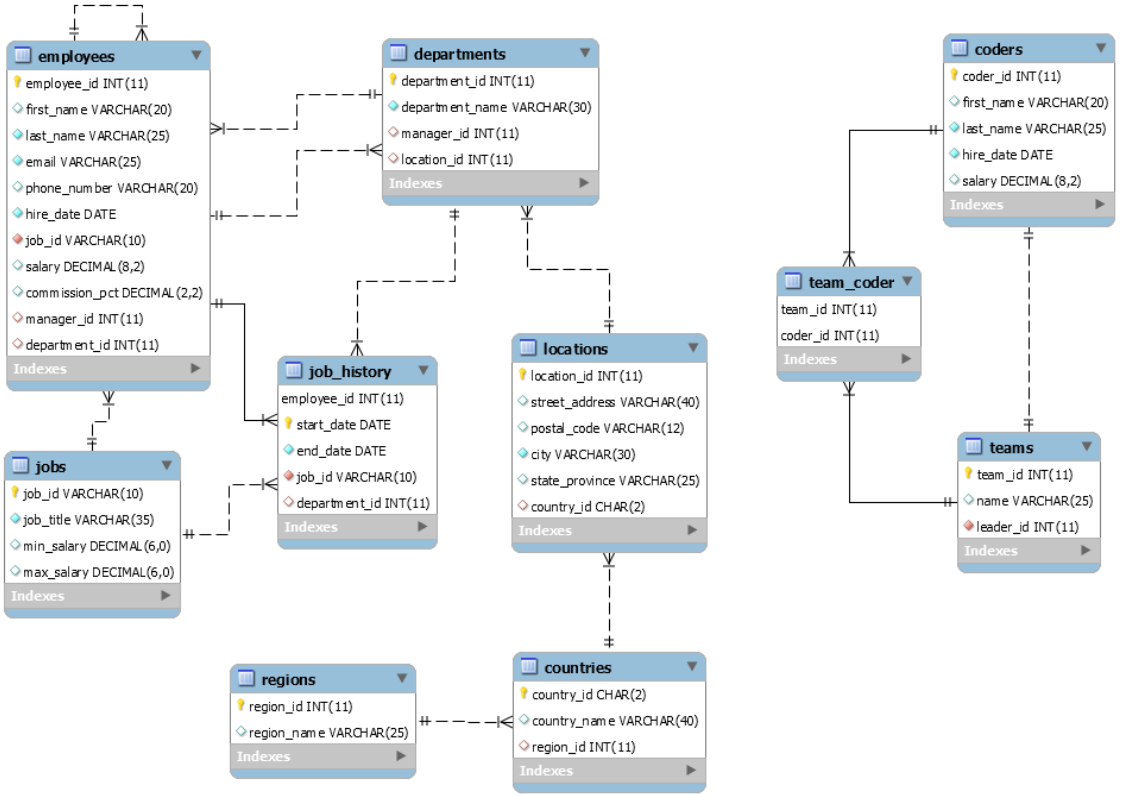
utente → connessione a MySql
schema → contenitore di oggetti di database

password delimitata da apici e case sensitive
in MySql *schema* è sinonimo di *database*
privilegi su oggetti di uno schema per utente
privilegi su procedure di uno schema per un utente

- L'utente root ha tutti i privilegi per la gestione dell'istanza corrente di MySql
- Altri comandi
  - use me; -- selezione del database correntemente in uso
  - show schemas; -- tutti gli schema disponibili all'utente corrente
  - source migration.sql -- per richiamare uno script da uno script (esecuzione via CLI)

# Diagramma Entity-Relation

MySQL Workbench
Database
Reverse Engineer...



# Principali tipi di dato

**DECIMAL**(precision, scale)

**INTEGER**, INT

CHAR(length)

**VARCHAR**(length)

**DATE**

TIMESTAMP

FLOAT, DOUBLE

In MySQL il confronto tra stringhe è per default *case insensitive*

# SELECT

- Selezione di dati (colonne) da una tabella, filtrata per colonne e righe

  select region_name from regions where region_id = 1;
- Selezione dei soli valori unici

  select distinct manager_id from employees;
- Modifica i risultati in lettura da tabella

  select job_title, min_salary, min_salary + 2000, min_salary * 3 + 1000 from jobs;
- Alias di colonna, introdotto da AS (opzionale) e delimitato da apici (singoli o doppi)

  select job_title, min_salary + 2000 "increased min salary" from jobs;
- La tabella DUAL (implicita e fittizia)

  select 1+2, 3-4, 2*6, 5/2, current_date -- from dual;
- Concatenazione

  select concat(country_id, "...", region_id, '!' ) from countries;
- Limitazione del numero di righe ritornate dalla query via clausola LIMIT

  select first_name, last_name from employees limit 1;

# Informazioni su tabelle e utenti

- Tabelle

  show tables; --  del database corrente

  select table_name from information_schema.tables; -- generale

  select * from information_schema.tables where table_schema='me';

- Descrizione di una tabella

  describe countries;

  select * from information_schema.columns c

  where c.table_schema='me' and c.table_name = 'countries';

- Descrizione degli utenti

  select * from mysql.user;

# NULL

- Valore non presente o non valido, check esplicito con “is null”

```
select first_name, last_name
from employees
where commission_pct is null;
```

- “Assorbe” altri operandi

```
select first_name, last_name, 12 * salary * commission_pct from employees;
```

- La funzione IFNULL() permette di decidere il comportamento

```
select first_name, last_name, 12 * salary * ifnull(commission_pct, 0)
from employees;
```

# Operatori di confronto

**=, !=, <, >, <=, >=**

```
select * from regions where region_id = 1;
select * from regions where region_id != 2;
select * from regions where region_id < 3;
select * from regions where region_id <= 3;
```

# Operatori di confronto

LIKE, BETWEEN, IN, IS NULL. Per negare il loro risultato: **NOT**

- **LIKE** wildcard: _ %

  select last_name from employees where last_name like '_ul%';
- **BETWEEN**

  select * from regions where region_id between 2 and 3;

  select * from countries where country_name between 'a' and 'c';
- **IN**

  select * from regions where region_id not in (2, 3);

  select * from regions where region_id not in (2, 3, null); -- !! NOT IN(…, NULL) → FALSE !!
- **IS NULL**

  select * from employees where manager_id is null;

In MySQL il confronto tra stringhe è per default *case insensitive*
cfr: LIKE **BINARY**

# Operatori logici

- **AND**

  ```
  select * from employees
  where salary < 3000 and employee_id > 195;
  ```

- **OR** (disgiunzione inclusiva)

  ```
  select * from employees
  where salary > 20000 or last_name = 'King';
  ```

- **NOT**

  ```
  select * from employees
  where not department_id > 20;
  ```

# Ordinamento via ORDER BY

- ORDER BY segue FROM – WHERE

```
select * from employees
order by last_name;
```

- ASC (ascending, default) / DESC (descending)

```
select * from employees
order by last_name desc, first_name asc;
```

- notazione posizionale

```
select first_name, last_name from employees
order by 2;
```

# Esercizi

- Employees: nome, cognome, email, telefono, data di assunzione
  - Tutti i dipendenti, ordinati per cognome e nome
  - Chi ha nome David o Peter
  - Chi appartiene al dipartimento 60
  - Chi appartiene ai dipartimenti 30, 50
  - Chi ha salario
    - maggiore di 10000
    - minore di 4000 o maggiore di 15000
    - minore di 4000 o maggiore di 15000, ma solo per i dipartimenti 50 e 80

# Esercizi

- Employees
  - Chi è stato assunto nel 2005
  - Quali job_id sono presenti, in ordine naturale
  - Chi ha una commissione
  - Chi ha una ‘a’ nel nome o cognome
- Departments
  - Nomi, in ordine naturale
- Locations
  - Indirizzi delle sedi italiane