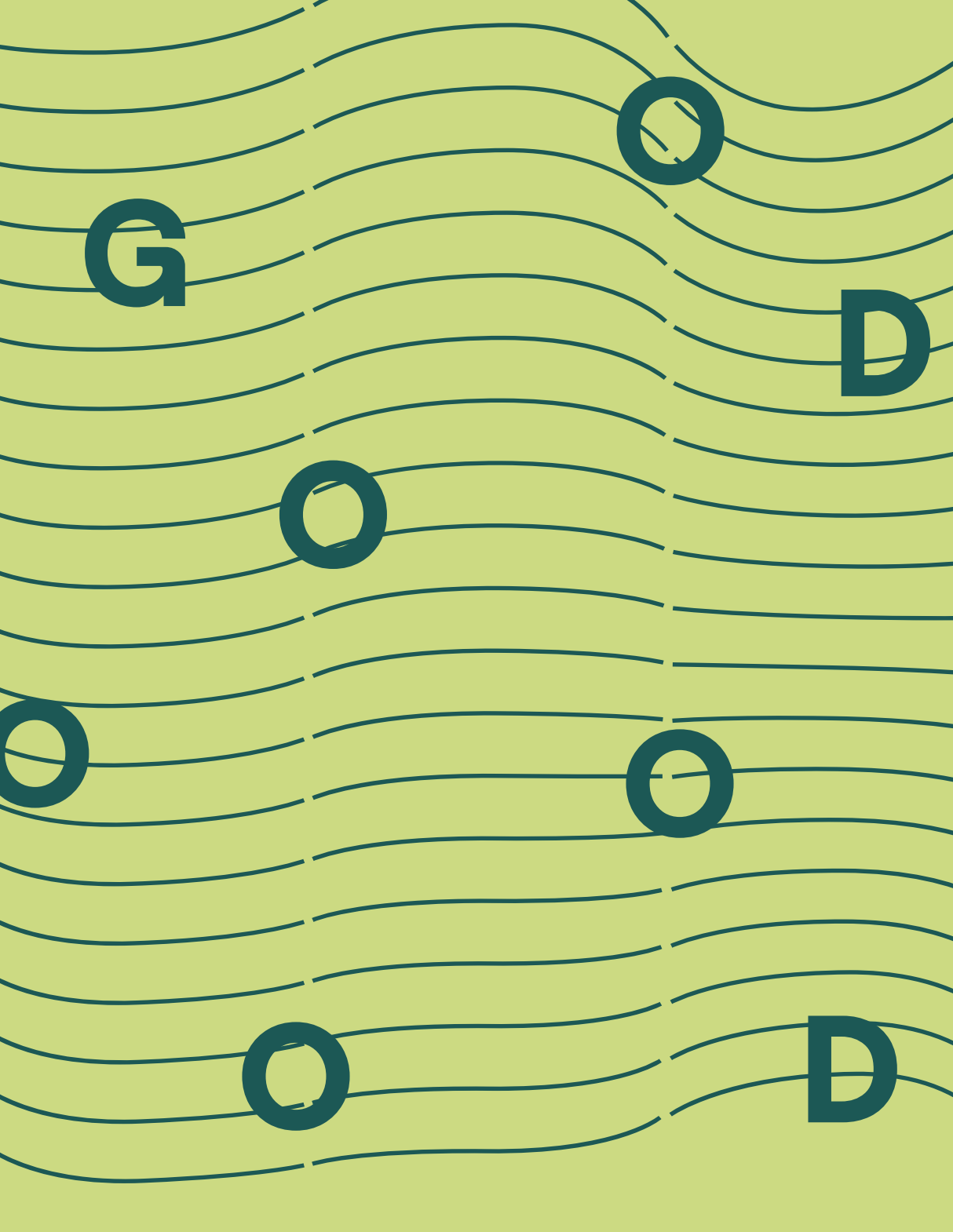

G
O
D
O
O
O
O
D

la caffettiera del masochista

insieme di dati che descrivono una realtà e che voglio interrogare

Database: an organized set of data that supports an carrying out specific activities (e.g. institution, enterprise, office)

Due approcci diversi: • Methodological

• technological

Contenuti: - Modello per organizzare / rappresentare i dati

- modo per interrogare la base di dati e x gestirla => LINGUACCIO

- sistemi x gestire i dati

- metodologie x realizzare / rappresentare una base di dati

INFORMATION SYSTEM: - componente di un'istituzione che gestisce informazioni

- ogni istituzione ha il proprio sistema informativo

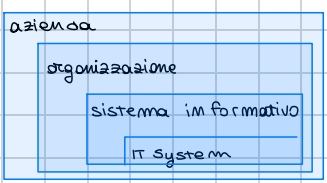
- supporta altri "subsystem" (per questo dovrebbe essere studiato dentro un contesto operativo)

GESTIONE INFORMAZIONI: - acquisire e memorizzare - elaboration, transformation, production

- conservare nel tempo - distribution, communication, exchange

└ alcune organizzazioni hanno il solo scopo di gestire i dati (es. banche)

IT SYSTEM parte digitalizzata/automatizzata del sistema informativo (utilizza tecnologie)



GESTIONE INFORMAZIONI:

rappresentate in modi diversi: - idee - disegni, diagrammi...

- linguaggio naturale - numeri e codici

H

IT-SYSTEM: l'informazione è espressa in dati

fatti forniti/imparati └ un valore grezzo che interpretato

a proposito di qualcosa o qualcuno mi consente di avere l'informazione

DATABASE (def. specifica): set di dati gestito da DBMS

└ struttura che consente di gestire le basi di dati (è equivalente del compilatore)

DATABASE MANAGEMENT SYSTEM:

gestisce collezione set di dati: - grandi - persistenti - condivisa

└ mentre assicura: - privacy - affidabilità - efficienza - efficacia

- la grandezza di un database è più grande della memoria principale, solo la grandezza fisica è il limite considerato

- Devono essere persistenti, la durata del database è indipendente da quella del computer

- " " condivisi, (es. nelle grandi aziende ogni area ha il proprio sottosistema informativo)

PROBLEMI - ridondanza (stesso dato appare + volte)/(stessa op. sullo stesso dato)

- inconsistenza

essendo condiviso: abbiamo più attività svolte sui dati condivisi: richiede permessi di accesso

più utenti hanno accesso: la base deve rimanere consistente (es. mele = -1 -> NO)

privacy: "utente A può leggere e editare x"

" " B " " y e " "

AFFIDABILITÀ: devono essere pronte a qualsiasi problema/imprevisto

TRANSAZIONI: insieme di op. eseguite insieme che lavorano sul DB consistente e lo lasciano consistente (anche dopo le modifiche)

se devo trasferire soldi da A a B, o vengono ritirati i soldi da A e versati su B o nessuna delle due op. viene eseguita."

CONCORRENZA: su un database lavorano + persone. PROB: le modifiche fatte da tante persone

ESERCIZI 2/10/24

es. 1

Region(Name, Surface)

City(Code, Name, Population)

Belong(Region, City)

- $\Pi_{region}(BELONG \bowtie_{city=code} \sigma_{Population>1M}(City)) \bowtie_{region=name} Region$

- $\Pi_{name}(P(Region \leftarrow Name(\sigma_{surface>2.5k}(REGION)) \bowtie BELONG \bowtie P_{city \leftarrow code}(CITY))$

es. 2

TRAIN(Code, Start, End, Miles)

$\sigma_{start='Boston' \wedge End='Chicago'}($

$P(CodeS, Switch, MilesS \leftarrow Code, End, Miles(TRAIN))$

$\bowtie$

$P(CodeE, Switch, MilesS \leftarrow Code, Start, Miles(TRAIN))$

oppure

$P(CodeS, Switch, MilesS \leftarrow Code, End, Miles(TRAIN))$

$\bowtie_{Switch=Switch \wedge Start='Boston' \wedge End='Chicago'}$

$P(CodeE, Switch, MilesS \leftarrow Code, Start, Miles(TRAIN))$

oppure

$$P\left(\text{CodeS, Switch, MilesS} \leftarrow \text{Code, Ends, Miles}\left(\sigma_{\text{Start}=\text{'Boston'}}(\text{TRAIN})\right)\right)$$

$$\bowtie_{\text{Switch}=\text{Switch}}$$

$$P\left(\text{CodeE, Switch, MilesS} \leftarrow \text{Code, Start, Miles}\left(\sigma_{\text{End}=\text{'Chicago'}}(\text{TRAIN})\right)\right)$$

es. 3

PERSON (Id , Name, Surname, Age)

ENROL (Person , Course)

COURSE (Name, Price)

$$\pi_{\text{Name}}\left(\sigma_{\text{Price} > \text{Age}*10}\left(\rho_{\text{person} \leftarrow \text{Id}}(\text{PERSON}) \bowtie \text{ENROL} \bowtie \rho_{\text{course} \leftarrow \text{Name}}(\text{COURSE})\right)\right)$$

09/10/24

- SAILOR (Id , Name , Level , BirthD)

  BOOKING (sailor , Boat, Date)

  BOAT (Id , Name, Colour)

  . Marinai che hanno prenotato una barca verde e una rossa

$$\pi_{\text{Id, Name}}\left(\text{Sailor} \bowtie_{\text{Id}=\text{sailor}} \left(\pi_{\text{sailor}}\left(\text{BOOKING} \underset{\text{boat}=\text{Id}}{\bowtie} \sigma_{\text{color}=\text{'red'}}(\text{BOAT})\right)\right) \bowtie \left(\pi_{\text{sailor}}\left(\text{BOOKING} \underset{\text{boat}=\text{Id}}{\bowtie} \sigma_{\text{color}=\text{'green'}}(\text{BOAT})\right)\right)\right)$$

- PLANE (Id , Name , Range)

  CERTIF (EId , PId)

  EMPLOYEE (Id , Name, Wage)

  . Id e Nome di piloti che possono volare su almeno 2 aeroporti che possono fare 5mila miglia

$$\pi_{\text{Id, Name}}(\text{EMPLOYEE}) \bowtie_{\text{Id}=\text{EId}} \left(\rho_{\text{EId1} \leftarrow \text{EId}\ \text{PId1} \leftarrow \text{PId}}\left(\left(\left(\pi_{\text{PId, EId}}\left(\text{CERTIF} \bowtie_{\text{PId}=\text{Id}} \pi_{\text{Id}}\left(\sigma_{\text{Range} \geq 5\text{mila}}(\text{PLANE})\right)\right)\right)\right)\right) \bowtie_{\text{EId1}=\text{EId} \wedge \text{PId} \neq \text{PId}} \left(\pi_{\text{PId, EId}}\left(\text{CERTIF} \bowtie_{\text{PId}=\text{Id}} \pi_{\text{Id}}\left(\sigma_{\text{Range} \geq 5\text{mila}}(\text{PLANE})\right)\right)\right)\right)$$

- SHOW (Code, Title, Comp , Duration)

  PLAYBILL (Date , Time, Show , Theatre)

  Comp che hanno fatto uno spett il 15/10 mon non il 16

$$\pi_{comp}(\pi_{Id,comp}(SHOW \bowtie_{code=Show} \theta_{Date=15/10}(PLAYBILL)) - \pi_{Id,Comp}(SHOW \bowtie_{code=Show} \theta_{16/10}(PLAYBILL)))$$

- EMPLOYEE(Code, Surmame, Wage, Dept.)

  DEPARTMENT(Id, Name, Location, Director)

  PROJECT(Num., Name, Budget, Manager)

  STAFF(Employee, Project)

  Nome imp. che lavorano a pj >100k

$$\pi_{Surmame}(EMPLOYEE \bowtie_{\substack{Code=\\employee}} (STAFF \bowtie_{\substack{Project\\=Num}} (\theta_{budget>100K}(PROJECT))))$$

Surmame impiegati che guadagnano + del manager

$$\pi_{surmame}(EMPLOYEE \bowtie_{\substack{Dept=Id\\ \wedge wage>w}} DEPARTMENT \bowtie_{\substack{director\\=c}} \rho_{C,S,W,D \leftarrow Code,Surname,Wage,Dept.}(Employee))$$

Surmame capi dept dei project manager

$$\pi_{surm.}((EMPLOYEE \bowtie_{\substack{Code\\=Director}} DEPARTMENT) \cup \pi_{surmame}(EMPLOYEE \bowtie_{\substack{code\\=manager}} PROJECT))$$

- nome dept dove wage.employee > 60K

$$\pi_{Nome}(\pi_{Id,Nome,Location,Director}(DEPARTMENT \bowtie_{Id=Dept} EMPLOYEE) - \pi_{Id,Nome,Location,Director}(EMPLOYEE \bowtie_{\substack{Dept=Id\\ \wedge\\ Wage<60K}} DEPT))$$

$$\pi_{surmame}(EMPLOYEE \bowtie_{\substack{Dept=Id\\ \wedge\\ Code=Director}} DEPARTMENT)$$

cognome impiegato con lo stipendi + alto

$$\pi_{code,surmame}(EMPLOYEE) - (\pi_{code,surmame}(EMPLOYEE \bowtie_{wage<w} \rho_{C,S,W,D \leftarrow Code,Surname,Wage,Dept}(EMPLOYEE))$$

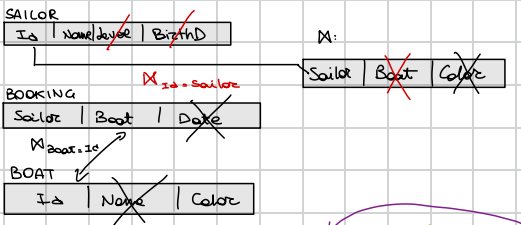
- SAILOR (Id, Name, Level, BirthD)

  BOOKING (Sailor, Boat, Date)

  BOAT (Id , Name, Color)

  • Marinai che hanno prenotato sia una barca verde che una rossa

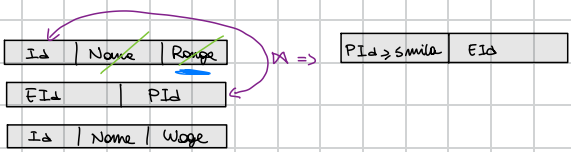
$$\pi_{Id, Nome}(SAILOR \bowtie_{Id=Sailor} (\pi_{Sailor, Boat}(BOOKING \bowtie_{Boat=Id} (\pi_{Id,Color}(\sigma_{Color='RED'}(BOAT))) \bowtie \pi_{Sailor, Boat}(BOOKING \bowtie_{Boat=Id} (\pi_{Id,Color}(\sigma_{Color='GREEN'}(BOAT)))))))$$



- PLANE (Id, Name, Range)

  CERTIF (EId, PId)

  EMPLOYEE (Id, Name, Wage)

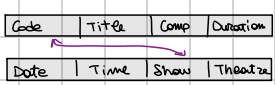


• Id e Name di piloti che possono volare su almeno 2 aeroplani che possono fare 5 mila miglia

$$\pi_{Id, Nome}(EMPLOYEE \bowtie_{Id=EId} (\rho_{PId1, EId1 \leftarrow PId, EId}(CERTIF \bowtie_{PId=Id} \pi_{Id}(\sigma_{Range \geq 5mila}(PLANE))) \bowtie_{EId1=EId \wedge PId1 \neq PId} (CERTIF \bowtie_{PId=Id} \pi_{Id}(\sigma_{Range \geq 5mila}(PLANE))))))$$

- SHOW(Code, Title, Comp, Duration)

  PLAYBILL (Date, Time, Show, Theatre)



• Comp che hanno fatto uno show il 15/10 ma il 16/10

$$\pi_{comp}(\pi_{Id, comp}(SHOW \bowtie_{Code=Show}(\sigma_{Date=15/10}(PLAYBILL))) - \pi_{Id, Comp}(SHOW \bowtie_{Code=Show}(\sigma_{Date=16/10}(PLAYBILL))))$$

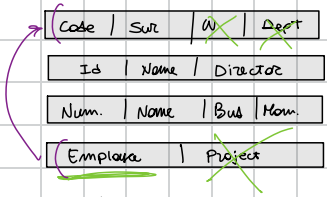
es. 28

- EMPLOYEE(Code, Surname, Wage, Dept)

  DEPT(Id, Name, location, Director)

  PROJECT (Number, Name, Budget, Manager)

  STAFF (Employee , Project)



Code e Surname degli impiegati che lavorano su più di un progetto

$$\pi_{code,surname}\Big(\big(\pi_{employee}\big(\rho_{E_1,P_1 \leftarrow employee,Project}(STAFF) \bowtie_{E_1=Employee \wedge P_1 \neq Project} (STAFF)\big)\big) \bowtie_{employee=code} \big(\pi_{code,surname}(EMPLOYEE)\big)\Big)$$

- EMPLOYEE(Code, Surname, Wage, Dept)

  DEPT(Id, Name, location, Director)

  PROJECT (Number, Name, Budget, Manager)

  STAFF (Employee , Project)

Code e Surname che lavorano a UN SOLO progetto

$$\pi_{code,surname}\Big(\big(\pi_{employee}(STAFF)\big) - \big(\pi_{employee}\big(\rho_{E_1,P_1 \leftarrow employee,Project}(STAFF) \bowtie_{E_1=Employee \wedge P_1 \neq Project} (STAFF)\big)\big)\Big) \bowtie_{employee=code} \big(\pi_{code,surname}(EMPLOYEE)\big)$$

ES. LIBRO

3.6) FILM (CodiceFilm, Titolo, Regista, Anno, CostoNoleggio)

ARTISTI (CodiceAttore, Cognome , Nome, Sesso, DataNascita, Nazionalità)

INTERPRETAZIONI (CodiceFilm, CodiceAttore, Personaggio)

1) Base di dati per la quale i join fra le varie relazioni siano tutti completi

$$\pi_{CodiceFilm,CodiceAttore,Personaggio,Cognome,Nome,Sesso,DataNascita,Nazionalità,Titolo,Regista,Anno,CostoNoleggio}\Big(\big(\big(\rho_{codF \leftarrow CodiceFilm}(FILM)\big) \bowtie_{codF=CodiceFilm} INTERPRETAZIONI\big) \bowtie \big(\big(\rho_{codA \leftarrow codiceAttore}(ARTISTI)\big) \bowtie_{codA=CodiceAttore} INTERPRETAZIONI\big)\Big)$$

- titoli film interpretati da Henry Fonda

$$\pi_{Titolo}\Big(\pi_{CodiceAttore,CodiceFilm}\big(\big(\rho_{CodiceAttore \leftarrow CodA}\big(\pi_{CodiceAttore}\big(\sigma_{Nome='Henry' \wedge Cognome='Fonda'}(ARTISTI)\big)\big)\big) \bowtie_{CodA=CodiceAttore} \big(\pi_{CodiceFilm,CodiceAttore}(INTERPRETAZIONI)\big)\big) \bowtie_{CodiceFilm=CodF} \big(\rho_{CodiceFilm \leftarrow CodF}\big(\pi_{CodiceFilm,Titolo}(FILM)\big)\big)\Big)$$

- Titoli dei film per i quali il regista è stato anche interprete

$$\pi_{Titolo}\left(\left(\pi_{Regista, CodiceFilm, Titolo}(FILM)\right) \bowtie_{Regista = Cognome} \left(\pi_{codiceAttore, Cognome}(ARTISTI)\right)\right)$$

- titolo dei film in cui gli attori mati siamo tutti dello stesso sesso. $M\text{-}MF \wedge F\text{-}MF$

$$FilmM := \left(\pi_{\substack{CodA,\\ CodiceFilm}}\left(\left(\rho_{\substack{CodiceAttore\\ \leftarrow CodA}}\left(\pi_{\substack{codice\\ Attore}}\left(\sigma_{Sesso='M'}(ART.)\right)\right)\right) \bowtie_{CodA = CodiceAttore} (INTERPRETAZIONI)\right)\right)$$

CodA	CodiceAttore	CodiceFilm	Personaggio

$$FilmF := \left(\pi_{\substack{CodA,\\ CodiceFilm}}\left(\left(\rho_{\substack{CodiceAttore\\ \leftarrow CodA}}\left(\pi_{\substack{codice\\ Attore}}\left(\sigma_{Sesso='F'}(ART.)\right)\right)\right) \bowtie_{CodA = CodiceAttore} (INTERPRETAZIONI)\right)\right)$$

$$\pi_{CodiceFilm}\left(FilmM \cap FilmF\right)$$

ESERCIZI SQL 18/10/24

1) LECTURE (Id, Surname, Dept)

STUDENT (Id, Surname)

COURSE (Code, Name)

EDITION (Course, Year, Lecturer)

EXAM (Student, Course, Year)

SELECT < list >
FROM
[WHERE]
[GROUP BY]
[HAVING]

Q: Cognomi comuni distinti fra studenti e docenti

- SELECT DISTINCT Surname
  FROM LECTURER
  INTERSECT
  SELECT DISTINCT Surname
  FROM STUDENT

- SELECT DISTINCT *
  FROM (SELECT Surname
  FROM LECTURER
  INTERSECT
  SELECT Surname
  FROM STUDENT)

meglio evitare le query annidate

- SELECT DISTINCT Surname
  FROM LECTURER, STUDENT
  WHERE LECTURER.Surname = STUDENT.Surname

- SELECT DISTINCT Surname
  FROM LECTURER JOIN STUDENT
  ON Surname = Surname

- Q: Cognome dei docenti che hanno corsi dove almeno 10 studenti hanno superato l'esame

- SELECT DISTINCT L.Surname
  FROM LECTURER AS L, EDITION AS E, EXAM AS X
  WHERE L.Id = E.lecturer AND E.Course = X.Course AND E.year = X.year
  GROUP BY L.Id, E.Course, E.Year, L.Surname
  HAVING COUNT(*) > 10

- SELECT L.Surname
  FROM LECTURER AS L JOIN EDITION AS E ON
  L.Id = E.lecturer NATURAL JOIN EXAM AS X

ES. 2

TRAIN (Code, Depart, Arriv, Mile) Q: Min e Max Mile tra Chicago e Boston senza combi

- SELECT MIN(TRAIN.Mile), MAX(TRAIN.Mile)
  FROM TRAIN
  WHERE TRAIN.Depart = 'Boston' AND
  TRAIN.Arriv = 'Chicago'

ES. 3

REGION (Name, Pop., Surf.)

RESIDENCE (Id, Surname, Region)

Q: Regioni che hanno più abitanti che residenti

- SELECT R.Name
  FROM REGION AS R JOIN RESIDENCE AS RE ON
  R.Name = Re.Region
  GROUP BY R.Name
  HAVING COUNT(*) > R.Pop

- SELECT Re.Name
  FROM REGION AS RE, (SELECT REGION, COUNT(*) AS citizen
  FROM RESIDENCE AS R
  GROUP BY R.Region) AS C
  WHERE Re.Name = C.Region AND RE.Pop > C.Citizen

es. 4

RATING (FilmId, UserId, Rating)

USER (UserId, Aeios, Age)

FILM (FilmId, Title, Year, Director)

Q: titolo film prodotti tra 90 e 2000 con AVG Rating ≥ 7

- SELECT FILM.Title
  FROM FILM NATURAL JOIN RATING
  WHERE FILM.Year > 1990 AND FILM.Year ≤ 2000
  GROUP BY FILM.FilmId, FILM.Title
  HAVING AVG(RATING.Rating) ≥ 7

Q: Min Age utenti che hanno valutato Blade Runner con almeno 8

SELECT MIN(User.Age)
FROM FILM NATURAL JOIN RATING NATURAL JOIN USER
WHERE FILM.Title = 'Blade Runner' AND RATING.Rating ≥ 8

23/10/24

SONG (Id, Title, Album, Plays, Length)

PLAYLIST (Name, Position, Song)

ALBUM (Id, Title, Year, Artist)

ARTIST (Id, Name, Labels)

Playlist con album prima del 2001 appartenenti UHA

```
SELECT  P.Name
FROM  PLAYLIST AS P  JOIN SONG AS S  ON P.SONG = S.Id
        JOIN ALBUM AS AL ON S.Album = AL.Id
        JOIN ARTIST AS A  ON  AL.Artist = A Id
WHERE  AL.Year < 2001  AND  A.Labels = 'UHA'
```

Playeist con canzoni pubblicate prima del 2001 di artisti appartenenti a 'UHA'.

ACTOR (Id, Name, Year, Nat)

RECITAL (Actor, Film)

FILM (Code, Title, Year, Director, Country, Genre)

SCREENING (Code, Film, Room, Date, Profits)

ROOM (Code, Name, Seats, City)

Q:

```
SELECT  Title
FROM  FILM
WHERE  Director = 'Feeini'  AND  year > 1960
```

Q: fantascienza giapponese dopo il 90 o fantascienza francese

```
SELECT  Title
FROM  FILM
WHERE  genre = 'Sci-Fi' AND ((country = 'Japan' AND Year > 1990) OR Country = 'France'))
```

Q: Titolo e genere Film proiettato a Londra le marcle scorso

```
SELECT DISTINCT F.Title, F. genre
FROM  FILM AS F  JOIN SCREENING AS S  ON  F.Code = S. Film
      JOIN ROOM AS R  ON  S.Room = R.Code
WHERE  R.City = 'London'  AND  S.Date = '25/12/23'
```

Q: Numero sale a Londra con + di 60 posti

```
SELECT  COUNT(*)
FROM  ROOM
WHERE    City = 'London'  AND    Seats > 60
```

Q: Coppie Attori-Registi che hanno lavorato assieme

```
SELECT A.Nome, F.Director, COUNT(*)
FROM ACTOR AS A, RECITAL AS R, FILM AS F
WHERE A.Id = R.Actor AND R.Film = F.Code
GROUP BY A.Id, F.Director, A.Nome
```

Q: Regista e Titolo di film che hanno meno di 6 Attori

```
SELECT F.Director, F.Title
FROM FILM AS F, RECITAL AS R
WHERE F.Code = R.Film
GROUP BY F.Director, F.Title, F.Code
HAVING COUNT(*) < 6
       /COUNT(R.Actor) < 6
```

per riuscire ad includere anche i film che hanno 0 attori: left join

```
SELECT F.Director, F.Title
FROM FILM AS F LEFT JOIN RECITAL AS R
    ON F.Code = R.Film
GROUP BY F.Director, F.Title, F.Code
HAVING COUNT(*) < 6
```

oppure

```
SELECT F.Director, F.Film
FROM FILM AS F
WHERE 6 > ( SELECT COUNT(*)
            FROM RECITAL AS R
            WHERE R.Film = F.Code)
```

Nome sala di proi. e incasso a Roma che a gennaio 2005 hanno guadagnato 20mila €.

```
SELECT R.Name, SUM(S.Profits)
FROM SCREENING AS S JOIN ROOM AS R ON S.Room = R.Code
WHERE R.City = 'Roma' AND S.Date ≥ '01/01/2005' AND S.Date ≤ '31/01/05'
GROUP BY R.Code, R.Name
HAVING SUM(S.Profits) > 20K
```

Q: Titoli film mai proiettati a Berlino

```
SELECT (F.Code, F.Title)
FROM FILM AS F
WHERE 'Berlin' NOT IN (SELECT DISTINCT R.City
                       FROM ROOM AS R JOIN SCREENING AS S
                           ON R.CODE = S.Room
                       WHERE S.Film = F.Code)
```

Q:

```
SELECT F.Code, F.Title
FROM FILM AS F
WHERE NOT EXIST ( SELECT *
                  FROM SCREENING AS S JOIN ROOM AS R ON S.Room = R.Code
                  WHERE F.Code = S.Film AND R.City = 'Berlin')
```

Titoli film che non hanno mai avuto incassi maggiori di 500€

```
SELECT F.Title
FROM FILM AS F
WHERE F.Code NOT IN (SELECT S.Film
      / NOT EXIST
                     FROM SCREENING AS S
                     WHERE S.Profits < 500 AND S.Film = F.Code)
```

```
/SELECT F.Title
 FROM FILM AS F JOIN SCREENING AS S
      ON F.Code = S.Film
 GROUP BY F.Code, F.Title
 HAVING MAX(S.Profits) < 500
```

Q: Film che hanno guadagnato almeno 500 in tutte le proiezioni

```
SELECT F.Title
FROM FILM AS F JOIN SCREENING AS S
     ON F.Code = S.Film
GROUP BY F.Code, F.Title
HAVING MIN(S.Profits) > 500
```

```
/SELECT F.Title
 FROM FILM AS F
 WHERE 500 ≤ (SELECT MIN(S.Profits)
              FROM SCREENING AS S
              WHERE F.Code = S.Film )
```

- MUSEUM(Nome, City)
  ARTIST (Name, Nation)
  WORK (Code, Title, NomeM, NomeA)
  CHARACTER(Nome, CodeW)

Nome Musei di Londra che non hanno opere di Tiziano

```
SELECT M.Nome
FROM MUSEUM AS M
WHERE M.City = 'London' AND NOT EXIST ( SELECT *
                                        FROM WORK AS W
                                        WHERE W.NomeA = 'Tiziano' AND M.Nome = W.NomeM )
```

Q: Nomi musei a Londra che hanno SOLO opere di Tiziano

```
SELECT M.Nome
FROM MUSEUM AS M
WHERE M.City = 'London' AND
      'Tiziano' = ALL ( SELECT W.NomeA
                        FROM WORK AS W
                        WHERE M.Nome = W.NomeM )
```

Q: Musei con almeno 20 opere di artisti italiani

```
SELECT W.NomeM
FROM ARTIST AS A JOIN WORK AS W ON A.Nome = W.NomeA
WHERE A.Nation = 'Italy'
GROUP BY W.NomeM
HAVING COUNT(*) > 20
```

- CARS(Plate, Brand, Power, Owner, CodAss)
  OWNERS(Code, Name, Residence)
  INSURANCE(CodeAss, Name, Hq)
  ACCIDENT(CodeAcc, Location, Date)
  INVCAR(CodeAcc, Plate, Amount)

- AUTO BMW O CON 120 CAVALLI...

```
SELECT C.Plate, O.Name
FROM CARS AS C JOIN OWNERS AS O ON C.Owner = O.Code
     JOIN INSURANCE AS I ON C.CodAss = I.CodeAss
WHERE ( C.Brand = 'BMW' OR C.Power > 120 ) AND I.Name = 'AVIVA'
```

Q: Ogni assicurazione NOME assicurazione e #AUTO assicurate

```
SELECT I.Name, I.Hq, COUNT(*)
FROM INSURANCE AS I JOIN CARS AS C
     ON I.CodeAss = C.CodeAss
GROUP BY I.Name, I.Hq, I.CodeAss
```

Q: Auto coinvolte in un incidente e restituisco: targa, nome, somma totale:

```
SELECT C.Plate, I.Nome, ...
FROM CARS AS C, INSURANCE AS I, INVCAR AS IC
WHERE C.CodeFisms = I.CodeFisms AND C.Plate = IC.Plate
GROUP BY IC.PLATE, I.NAME
HAVING COUNT(*) > 1
```

SIMULAZIONE D'ESAME

MODELLO (IdModello, IdMarca, Nome, Categoria)

VALUTAZIONE (IdModello, Anno, Prezzo)

MARCA (IdMarca, Marchio, Nazionalità)

Q: Media delle valutazioni per ogni modello di auto italiana indicando anche la marca

```
SELECT M.Marchio, AVG(V.Prezzo)
FROM MARCA AS M JOIN MODELLO AS MO ON M.IdMarca = MO.IdMarca
     JOIN VALUTAZIONE AS V ON MO.IdModello = V.IdModello
WHERE M.Nazionalità = 'Italy'
GROUP BY V.IdModello, M.IdMarca
```

Q: Restituire, per ogni anno dal 2000 al 2010, il conteggio delle berline tedesche con valutazione maggiore di 30000.

```
SELECT COUNT(*), V.Anno
FROM MARCA AS M JOIN MODELLO AS MO ON M.IdMarca = MO.IdMarca
     JOIN VALUTAZIONE AS V ON MO.IdModello = V.IdModello
WHERE M.Nazionalità = 'Germania' AND MO.Categoria = 'Berlina' AND V.Anno > 2000 AND V.Anno < 2010 AND V.Prezzo > 30000
GROUP BY V.Anno
```

Q: Tornare le valutazioni delle berline prodotte dalle case francesi che non hanno valutazione per l'anno 2007

$$\pi_{IdModello,\ Prezzo,\ Anno}\left(\sigma_{Nazionalità='Francia' \wedge Categoria='Berlina'}\left((MARCA \bowtie MODELLO) \bowtie \left(VALUTAZIONE - \left(\sigma_{Anno='2007' \text{ AND } Prezzo \neq NULL}(VALUTAZIONE)\right)\right)\right)\right)$$

Q: Restituire le valutazioni superiori 15000, dall'anno 2005 al 2009, delle vetture non italiane

$$\pi_{IdModello,\ prezzo,\ anno}\left(\sigma_{Prezzo>15K \text{ AND } 2005<Anno<2009 \text{ AND } categoria='utilitaria' \text{ AND } Nazionalità \neq 'Italia'}(MARCA \bowtie MODELLO \bowtie VALUTAZIONE)\right)$$

es. 3.17

- $\Pi_{\text{NomeFarmaco, Nome, Codice}}$ (FARMACI ⋈ PRODUTTORI ⋈ ($\sigma_{\text{categoria='sulfamidico'}}$ (SOSTANZE))

- $\Pi_{\text{NomeFarmaco, NomeSostanza, Codice}}$ (FARMACI ⋈ ($\sigma_{\text{Nazione='Italia'}}$ (PRODUTTORI)) ⋈ (SOSTANZE))

ES.

STUDENT (StudentId, Name, Age, MajorId)

COURSE (CourseId, CourseName, Department)

ENROLLMENT (StudentId, CourseId, Semester, Grade)

MAJOR (MajorId, MajorName, Department)

PROFESSOR (ProfessorId, Name, Department)

TEACHES (ProfessorId, CourseId, Semester)

IO:

$\Pi_{\text{nameP, ProfessorId}}$ ($\rho_{\text{Dep, NomeP}} \leftarrow \text{Department, Nome}$ (PROFESSOR ⋈ TEACHES ⋈ ENROLLMENT ⋈ MAJOR) ⋈ $_{\text{Dep=Department}}$ (STUDENT ⋈ MAJOR))

SANJO:

$\Pi_{\text{name, ProfessorId}}$ ((PROFESSOR ⋈ TEACHES ⋈ ENROLLMENT ⋈ MAJOR) ⋈ $\Pi_{\text{department}}$ (STUDENT ⋈ MAJOR))

Q: studenti che sono iscritti ad almeno un corso del CS dept

$\Pi_{\text{StudentId}}$ (STUDENT ⋈ ENROLLMENT ⋈ ($\sigma_{\text{Department='Computer Science'}}$ (COURSE)))

Q: Nomi degli studenti che hanno ricevuto una A in un corso tenuto da un prof del dip. di Storia

STUDENT (StudentId, Name, Age, MajorId)

COURSE (CourseId, CourseName, Department)

ENROLLMENT (StudentId, CourseId, Semester, Grade)

MAJOR (MajorId, MajorName, Department)

PROFESSOR (ProfessorId, Name, Department)

TEACHES (ProfessorId, CourseId, Semester)

$$\Pi_{StudentId, Name}\Big(\big(\Pi_{ProfessorId}(\sigma_{Department='History'}(PROFESSOR))\big) \bowtie \Pi_{ProfessorId, CourseId}(TEACHES) \bowtie \Pi_{CourseId, StudentId}(\sigma_{Grade='A'}(ENROLLMENT)) \bowtie \big(\Pi_{StudentId, Name}(STUDENT)\big)\Big)$$

ES.5 Q: Id e nome degli studenti iscritti ai corsi tenuti dai prof del dept. di fisica

$$\Pi_{StudentId, Name}\Big(\big(\Pi_{ProfessorId}(\sigma_{Department='Physics'}(PROFESSOR))\big) \bowtie \big(\Pi_{ProfessorId, CourseId}(TEACHES)\big) \bowtie \big(\Pi_{CourseId, StudentId}(ENROLLMENT)\big) \bowtie \big(\Pi_{StudentId, Name}(STUDENT)\big)\Big)$$

ESERCITAZIONE 28/10 ALGEBRA

ES. 3

TRAIN (Code, Start, End, Miles)

Q: tutti i treni tra Boston e Chicago con un cambio

$$\rho_{Code1, Switch, Mile1 \leftarrow Code, End, Mile}(TRAIN) \bowtie_{Start = 'Boston' \wedge End = 'Chicago' \wedge switch = switch} \rho_{Start \leftarrow switch}(TRAIN)$$

ES. 4

PERSON (Id, Name, Surname, Age)

ENROLLMENT (Person, Course)

COURSE (Name, Price)

Q: nomi distinti delle persone che seguono un corso che costa più di \$10 per ogni anno della loro età

$$\Pi_{Name}\left(\sigma_{Price > Age \cdot 10}\left(\left(\left(\Pi_{Id, Name, Age}(PERSON)\right) \bowtie_{Id = Person} ENROLLMENT\right) \bowtie_{course = name} (COURSE)\right)\right)$$

ES. 5

STUDENT (Id, Name, Surname, School)

EXAM (Student, Subject, Mark, Date)

Q: Nome e Cognome degli studenti che hanno preso 30 in almeno un esame

$$\Pi_{Name, Surname}\left(\Pi_{Student}\left(\sigma_{MARK = 30}(EXAM)\right) \bowtie_{Student = Id} \left(\Pi_{Id, Name, Surname}(STUDENT)\right)\right)$$

Q: Nome e Cognome degli studenti che non hanno mai preso 30

$$\pi_{Id, Nome, Surname}(STUDENT)$$

$$\pi_{Id, Nome, Surname}(STUDENT \bowtie_{Id=Student}(\sigma_{Grade=30}(EXAM)))$$

ES. 7

SUPPLIER (Id, Name, Partners, Headquarter)
PRODUCT (Id, Name, Colour, Size, Store)
SUPPLYING (SupId, ProdId, Quantity)

Q: nome dei fornitori che forniscono almeno un prodotto oltre al 'PX274'

$$\pi_{Name, Id}(\hat{\pi}_{ProdId, SupId}(((\rho_{SupId1, ProdId1 \leftarrow SupId, ProdId}(\sigma_{ProdId \neq 'PX274'}(SUPPLYING))) \bowtie_{SupId1=SupId \wedge ProdId1 \neq ProdId} SUPPLYING)) \bowtie_{SupId=Id} (SUPPLIER))$$

TROPPO COMPLICATO ↑

↓

$$\Upsilon_{nome}\left(SUPPLIER \bowtie_{Id=SupId} \left(\sigma_{ProdId \neq 'PX274'}(SUPPLYING)\right)\right)$$

Q: codice dei fornitori che forniscono almeno due prodotti diversi

$$\Uparrow_{SupId}\left(\left(\left(\rho_{S1,P1 \leftarrow SupId, ProdId}(SUPPLYING)\right) \bowtie_{S1=SupId \wedge P1 \neq ProdId} SUPPLYING\right)\right)$$

ES. 9

BOOK (BookId, Title, Author)
USER (UserId, Name, Surname)
BORROWING (UserId, BookId, Date)

Q: Id degli user che hanno preso un libro di Fleming

$$\Uparrow_{UserId}\left(\left(\sigma_{Author='Fleming'}(BOOK)\right) \bowtie BORROWING\right)$$

EMPLOYEE(Code, Surname, Wage, Department)
DEPARTMENT(Id, Name, Location, Director)
PROJECT(Number, Name, Budget, Manager)
STAFF(Employee, Project)

Write a relational algebra expression that returns the surname of employees who earn more than their department head.

cognome dei dipendenti che guadagnano più dei loro capi dipartimento

$$HeadWage := \Uparrow_{Code, Wage, Id} (DEPARTMENT \bowtie_{Director = Code} EMPLOYEE)$$

$$\Upsilon_{Surname} ((\rho_{wageE \leftarrow Wage} (EMPLOYEE)) \bowtie_{DepE = Id \wedge wageE > wage} (HeadWage))$$

EMPLOYEE(Code, Surname, Wage, Department)
DEPARTMENT(Id, Name, Location, Director)
PROJECT(Number, Name, Budget, Manager)
STAFF(Employee, Project)

Write a relational algebra expression that returns the surnames of the department heads and project managers.

cognome dei capi dipartimento e dei project manager

$$SurnameManager := \Uparrow_{code, surname} (PROJECT \bowtie_{Manager = Code} EMPLOYEE)$$

$$SurnameDirector := \Upsilon_{code, surname} (DEPARTMENT \bowtie_{Director = Code} EMPLOYEE)$$

$$\Uparrow_{surname} (SurnameManager \cup SurnameDirector)$$

EMPLOYEE(Code, Surname, Wage, Department)
DEPARTMENT(Id, Name, Location, Director)
PROJECT(Number, Name, Budget, Manager)
STAFF(Employee, Project)

Write a relational algebra expression that returns the names of departments where all employees earn more than 60K.

Nomi dei dipartimenti dove tutti gli impiegati guadagnano più di 60 K

$$\pi_{nome}\Big(\big(\pi_{Department,nome}\big(\rho_{Department \leftarrow Id}(DEPARTMENT)\big)\big) - \big(\pi_{Department,nome}\big(\sigma_{wage<60K}\big((EMPLOYEE) \bowtie_{Department=Id} (DEPARTMENT)\big)\big)\big)\Big)$$

EMPLOYEE(Code, Surname, Wage, Department)
DEPARTMENT(Id, Name, Location, Director)
PROJECT(Number, Name, Budget, Manager)
STAFF(Employee, Project)

Write a relational algebra expression that returns the code and surname of employees not working on any project.

Codice e cognome impiegati che non lavorano ad alcun progetto

$$\pi_{Code,Surname}(STAFF) - \pi_{code,surname}(EMPLOYEE \bowtie_{code=Employee} STAFF)$$

EMPLOYEE(Code, Surname, Wage, Department)
DEPARTMENT(Id, Name, Location, Director)
PROJECT(Number, Name, Budget, Manager)
STAFF(Employee, Project)

Codice e nome degli impiegati che lavorano a più di un progetto

Write a relational algebra expression that returns the code and surname of employees working on more than one project.

$$\Pi_{Code,Surnome}\left(\left(\left(\rho_{E1,P1 \leftarrow Employee,Project}(STAFF)\right) \bowtie_{E1=Employee \wedge P1 \neq Project} STAFF\right) \bowtie EMPLOYEE\right)$$

EMPLOYEE(Code, Surname, Wage, Department)
DEPARTMENT(Id, Name, Location, Director)
PROJECT(Number, Name, Budget, Manager)
STAFF(Employee, Project)

Codice e cognome di impiegati che lavorano su un solo progetto

Write a relational algebra expression that returns the code and surname of the employees working on a single project.

$$\Pi_{code,surnome}\Big(\big(STAFF - \big(\big(\Pi_{Employee,Project}\big(\rho_{E1,P1 \leftarrow Employee,Project}(STAFF)\big) \bowtie_{E1=Employee \wedge P1 \neq Project} STAFF\big)\big)\big) \bowtie_{Employee=Code} EMPLOYEE\Big)$$

EMPLOYEE(Code, Surname, Wage, Department)
DEPARTMENT(Id, Name, Location, Director)
PROJECT(Number, Name, Budget, Manager)
STAFF(Employee, Project)

Cognome dell'impiegato più pagato

Write a relational algebra expression that returns the surname of the highest paid employee.

$$\tilde{\pi}_{code,\ surname}\left(\left(\rho_{w1 \leftarrow wage}(EMPLOYEE)\right) \bowtie_{w1 > wage} EMPLOYEE\right)$$

EMPLOYEE(Code, Surname, Wage, Department)
DEPARTMENT(Id, Name, Location, Director)
PROJECT(Number, Name, Budget, Manager)
STAFF(Employee, Project)

Codice e cognome impiegati che lavorano ad un solo progetto

Write a relational algebra expression that returns the code and surname of the employees working on a single project.

$$\tilde{\pi}_{code,\ surname}\left(\pi_{code}(EMPLOYEE) - \left(\rho_{code \leftarrow employee}\left(\pi_{employee}\left(\left(\rho_{E,P \leftarrow Employee,Project}(STAFF)\right) \bowtie_{E=Employee \wedge P \neq Project} STAFF\right)\right)\right)\right) \bowtie EMPLOYEE)$$

ESAME 1

M.1

nomi film con media valutazione superiore a 7, tra 1990 e 2000

```
SELECT F.TITOLO, F.CODFILM
FROM FILM AS F JOIN VOTO AS V
WHERE F.ANNO >= 1990 AND F.ANNO <= 2000
GROUP BY F.TITOLO, F.CODFILM
HAVING AVG(V.VALUTAZIONE) >= 7
```

M.2) età minima degli utenti che hanno dato a 'Blade Runner' un voto maggiore di 8

```
SELECT MIN(U.ETA')
FROM FILM AS F NATURAL JOIN VOTO AS V
     NATURAL JOIN UTENTE AS U
WHERE F.TITOLO = 'Blade Runner'
      AND V.VALUTAZIONE > 8
```

M.3) valutazione dei film dal 2000 in poi per l'utente AntonioBianchi

$$\Pi_{\text{VALUTAZIONE, CODFILM}}\left(\left(\sigma_{\text{ANNO}>=2000}(\text{FILM})\right) \bowtie \text{VOTO} \bowtie \left(\sigma_{\text{ALIAS}='\text{AntonioBianchi}'}(\text{UTENTE})\right)\right)$$

M.4) dati degli utenti che hanno votato Django ma non hanno votato Pulp Fiction

$$\Pi_{\text{CODUTENTE, ALIAS, ETA'}}\left(\text{UTENTE} \bowtie \left(\left(\sigma_{\text{TITOLO}='\text{Django}'}(\text{FILM})\right) \bowtie \text{VOTO} - \left(\sigma_{\text{TITOLO}='\text{Pulp Fiction}'}(\text{FILM})\right) \bowtie \text{VOTO}\right)\right)$$

ESAME 2

ES. 1) per ogni serie tv, tornare il nome della serie e la durata media degli episodi della prima stagione

```
SELECT S.NOME , AVG(E.DURATA)
FROM SERIE AS S JOIN EPISODIO AS E ON S.ID = E.IDSERIE
WHERE E.STAGIONE = 1
GROUP BY S.ID, S.NOME
```

ES.2) restituire i nomi delle prime 3 serie, create nel 2008, con il maggior numero di episodi totali

```
SELECT S.TITOLO
FROM SERIE AS S JOIN EPISODIO AS E ON S.ID = E.IDSERIE
WHERE S.ANNO = 2008
GROUP BY S.ID, S.TITOLO
HAVING COUNT(*) > (SELECT COUNT(*)
                   FROM SERIE AS S1 JOIN EPISODIO AS E
                        ON S1.ID = E.IDSERIE
                   WHERE S1.ANNO = 2008 AND S1.ID ≠ S.ID
                   GROUP BY S.ID)
```

ES. 3) nome delle serie tv che non hanno nemmeno un sottotitolo in italiano

$$\Pi_{NOME,ID}(SERIE) - \Pi_{NOME,ID}(\rho_{ID \leftarrow IDE}((SERIE \bowtie_{ID = IDSERIE} (\rho_{IDE \leftarrow ID}(EPISODIO)) \bowtie_{IDE = IDEPISODIO} (\sigma_{LINGUA='ITALIANO'}(SOTTOTITOLO)))))$$

ES. 4) restituire numero e stagione degli episodi della serie TWD che hanno durata superiore a 60 minuti per i quali esistono i sottotitoli in inglese

$$\pi_{\text{NUMERO, STAGIONE, IDE}}\Big(\big(\sigma_{\text{NOME}=\text{'The walking Dead'}}(\text{SERIE})\big) \bowtie_{\text{ID}=\text{IDSERIE}} \big(\rho_{\text{IDE}\leftarrow\text{ID}}\big(\sigma_{\text{DURATA}>60}(\text{EPISODIO})\big)\big) \bowtie_{\text{IDE}=\text{IDEPISODIO}} \big(\sigma_{\text{LINGUA}=\text{'INGLESE'}}(\text{SOTTOTITOLO})\big)\Big)$$

ESAME 2

es. 1: attributi dei concerti che contengono un movimento 'Largo'

```
SELECT C.RV, C.TONALITA', C.NOME, C.DATA
FROM MOVIMENTI AS M NATURAL JOIN CONCERTI AS C
WHERE M.TEMPO = 'Largo'
```

es. 2: attributi dei movimenti dei concerti scritti dopo il 1720 che contengono più di 3 movimenti

```
SELECT C.RV, C.TONALITA', C.NOME, C.DATA
FROM MOVIMENTI AS M NATURAL JOIN CONCERTI AS C
WHERE C.DATA > 1720
GROUP BY C.RV, C.TONALITA', C.NOME, C.DATA
HAVING MAX(M.NUMERO) > 3
```

es. 3: tutti i movimenti dei concerti dove compare almeno un violino che durano più di due minuti

$$\pi_{RV, NUMERO, TEMPO, DURATA}\left(\left(\sigma_{STRUMENTO='VIOLINO'}(STRUMENTAZIONE)\right) \bowtie \left(\sigma_{DURATA>120}(MOVIMENTI)\right)\right)$$

es. 4: concerti dove tra gli strumenti compare il cembalo e hanno solo tre movimenti

$$\left(\pi_{RV}\left(\left(\sigma_{STRUMENTO='CEMBALO'}(STRUMENTAZIONE)\right) \bowtie \left(\left(\sigma_{NUMERO=3}(MOVIMENTI)\right) - \left(\sigma_{NUMERO>3}(MOVIMENTI)\right)\right)\right)\right) \bowtie CONCERTI$$

ESAME 4

es. 1: restituire tutte le famiglie con il numero minore di millesimi

$$\text{FAMIGLIA} - \pi_{\text{VIA,NUMERO,INTERNO,FAMIGLIA,MILLESIMI}}\left(\left(\rho_{M \leftarrow \text{MILLESIMI}}(\text{FAMIGLIA})\right) \bowtie_{M < \text{MILLESIMI}} (\text{FAMIGLIA})\right)$$

es. 2: restituire le famiglie che non hanno spese personali di tipo luce

$$\text{FAMIGLIA} \bowtie \left(\pi_{\text{VIA,NUMERO,INTERNO}}(\text{SPESEPERSONALI} - (\sigma_{\substack{\text{TIPOBOLLETTA} = \text{'LUCE'} \\ \wedge \\ \text{CONSUMO} > 0}}(\text{SPESEPERSONALI})))\right)$$

es. 3: query che associa ad ogni famiglia le spese personali complessive

```
SELECT F.VIA, F.NUMERO, F.INTERNO, SUM(SP.CONSUMO * SP.TARIFFA)
FROM FAMIGLIA AS F LEFT JOIN SPESEPERSONALI AS SP
WHERE SP.CONSUMO IS NOT NULL AND SP.TARIFFA IS NOT NULL
GROUP BY F.VIA, F.NUMERO, F.INTERNO
```

es. 4: query che calcola la somma dei millesimi associati ad ogni numero di scala con almeno una famiglia

```
SELECT S.SCALA, SUM (F.MILLESIMI)
FROM FAMIGLIA AS F JOIN SCALA AS S
     ON F.INTERNO = S.INTERNO
GROUP BY S.SCALA
```

ESERCIZI SLIDE 29/10/24

p.46

es. 1: nome dei musei a Londra che non hanno opere di Tiziano

```
SELECT M.NAME
FROM MUSEUM AS M
WHERE M.CITY = 'London'
      AND NOT EXISTS (SELECT *
                      FROM WORK AS W
                      WHERE M.NAMEA = 'Tiziano'
                            AND M.NAME = W.NAMEM)
```

es. 2: nomi dei musei a Londra che hanno solo opere di Tiziano

```
SELECT M.NAME
FROM MUSEUM AS M
WHERE M.NAME = 'London'
      AND 'Tiziano' = ALL (SELECT W.NAMEA
                           FROM WORK AS W
                           WHERE M.NAME = W.NAMEM)
```

es. 3: nomi dei musei che hanno almeno 20 opere di artisti italiani

```
SELECT W.NAMEM
FROM WORK AS W JOIN ARTIST AS A ON W.NAMEA = A.NAME
WHERE A.NATIONALITY = 'Italy'
GROUP BY W.NAMEM
HAVING COUNT(*) >= 20
```

EMPLOYEE(Code, Surname, Wage, Department)
DEPARTMENT(Id, Name, Location, Director)
PROJECT(Number, Name, Budget, Manager)
STAFF(Employee, Project)

Write a relational algebra expression that returns the names of the projects with a budget greater than 100K and the surnames of the employees who work on them.

Nome progetti con un budget maggiore di 100K e il cognome di chi ci lavora

$$\pi_{Nome, Surnome}((\sigma_{Budget > 100K}(PROJECT)) \bowtie_{Number = Project} STAFF) \bowtie_{Employee = Code} EMPLOYEE)$$

EMPLOYEE(Code, Surname, Wage, Department)
DEPARTMENT(Id, Name, Location, Director)
PROJECT(Number, Name, Budget, Manager)
STAFF(Employee, Project)

Write a relational algebra expression that returns the code and surname of employees not working on any project.

codice e cognome degli impiegati che non lavorano ad alcun progetto

$$EMSTF := EMPLOYEE \bowtie_{Code = Employee} STAFF$$

$$\pi_{Code, Surnome}(EMPLOYEE - (\pi_{Code, Surnome, wage, Department}(EMSTF)))$$

EMPLOYEE(Code, Surname, Wage, Department)
DEPARTMENT(Id, Name, Location, Director)
PROJECT(Number, Name, Budget, Manager)
STAFF(Employee, Project)

Codice e Cognome degli impiegati che lavorano a più progetti

Write a relational algebra expression that returns the code and surname of employees working on more than one project.

$$EMSTF := EMPLOYEE \bowtie_{Code = Employee} STAFF$$

$$\left(\Pi_{code, surname}\left(\left(\rho_{c,p \leftarrow code, project}\left(\left(\Pi_{code, surname, project}(EMSTF)\right)\right)\right) \bowtie_{c = code \wedge p \neq project} \left(\Pi_{code, surname, project}(EMSTF)\right)\right)\right)$$

EMPLOYEE(Code, Surname, Wage, Department)
DEPARTMENT(Id, Name, Location, Director)
PROJECT(Number, Name, Budget, Manager)
STAFF(Employee, Project)

Codice e Cognome delle persone che lavorano su un solo progetto

Write a relational algebra expression that returns the code and surname of the employees working on a single project.

$$\Pi_{code, surname}(EMPLOYEE) - \left(\Pi_{code, surname}\left(\left(\rho_{c,p \leftarrow code, project}\left(\left(\Pi_{code, surname, project}(EMSTF)\right)\right)\right) \bowtie_{c = code \wedge p \neq project} \left(\Pi_{code, surname, project}(EMSTF)\right)\right)\right)$$

SLIDE ALGEBRA

p.30

es.8: codice dei fornitori che forniscono almeno due prodotti diversi

$$\Pi_{SupId}\left(\left(\rho_{S,P \leftarrow SupId,ProdId}(SUPPLYING)\right) \bowtie_{S=SupId \wedge P \neq ProdId} SUPPLYING\right)$$

p.46

es.13: nome delle classi che hanno 100 posti o più che non sono prenotate oggi

$$\Pi_{name}\left(\left(\sigma_{capacity>=100}(ROOM)\right) \bowtie_{code=Room} \left(\sigma_{Date <>' 29/10/24'}(BOOKING)\right)\right)$$

p. 54 n. 17

nomi delle pizzerie frequentate solo da maschi o solo da femmine

$$\pi_{Pizzeria}((\sigma_{gender='Male'}(PERSON)) \bowtie FREQUENTS) - \pi_{Pizzeria}((\sigma_{gender='Female'}(PERSON)) \bowtie FREQUENTS) \cup \pi_{Pizzeria}((\sigma_{gender='Female'}(PERSON)) \bowtie FREQUENTS) - \pi_{Pizzeria}((\sigma_{gender='Male'}(PERSON)) \bowtie FREQUENTS)$$

ESAME 3 REMAKE

es. 1

```
SELECT C.RV, C.TONALITÀ, C.NOME, C.DATA
FROM MOVIMENTI AS M NATURAL JOIN CONCERTI AS C
WHERE M.Tempo = 'Largo'
```

es. 2

```
SELECT C.*
FROM CONCERTI AS C NATURAL JOIN MOVIMENTI AS M
WHERE C.DATA > 1720
GROUP BY C.RV, C.TONALITÀ, C.NOME, C.DATA
HAVING COUNT(M.NUMERO) > 3
```

```
SELECT M.*
FROM CONCERTI AS C
     NATURAL JOIN MOVIMENTI AS M
WHERE C.DATA > 1720
      AND C.RV IN ( SELECT MO.RV
                    FROM MOVIMENTI AS MO
                    GROUP BY MO.RV
                    HAVING COUNT(M.NUMERO) > 3 )
```

es. 3

$$\pi_{RV, Numero, Tempo, Durata}\left(\sigma_{Strumento = \text{'Violino'}}(STRUMENTAZIONE)\right) \bowtie \left(\sigma_{Durata > 120}(MOVIMENTI)\right)$$

es.4

$$\pi_{\text{RV, Tonalità, Nome, Data}} \Big(\text{CONCERTI} \bowtie \Big(\big(\sigma_{\text{Strumento}='\text{Cembalo}'} (\text{STRUMENTAZIONE}) \big) \bowtie \big((\sigma_{\text{Numero}=3} (\text{MOVIMENTI})) - (\sigma_{\text{Numero}>3} (\text{MOVIMENTI})) \big) \Big) \Big)$$

$$\pi_{Nome, Nazione}(GIOCATORE) - \pi_{Nome, Nazione}(VITBIANCHI \cup VITNERI)$$

$$VITBIANCHI := (GIOCATORE \bowtie_{IdScacchista = GiocatoreBianco} (\sigma_{VITTORIA='BIANCO'}(PARTITA)))$$

$$VITNERI := (GIOCATORE \bowtie_{IdScacchista = GiocatoreNero} (\sigma_{VITTORIA='NERO'}(PARTITA)))$$

$$AK := ((\sigma_{Nome='AK'}(GIOCATORE)) \bowtie_{IdScacchiere = GiocatoreBianco \vee IdScacchiere = GiocatoreNero} (PARTITA))$$

$$NS := \left(\left(\sigma_{Nome = 'NS'}(GIOCATORE)\right) \bowtie_{IdScacchiere = GiocatoreBianco \vee IdScacchiere = GiocatoreNero} (PARTITA)\right)$$

$$\left(\left(\rho_{IdP \leftarrow IdPartita}(AK)\right) \bowtie_{IdP = IdPartita} (NS)\right)$$

ESERCIZI 22/11/24

gestione visite mediche, svolte in diverse cliniche, ogni visita e' svolta in una sola clinica

- Code
- indirizzo
- telefono

ENTITÀ

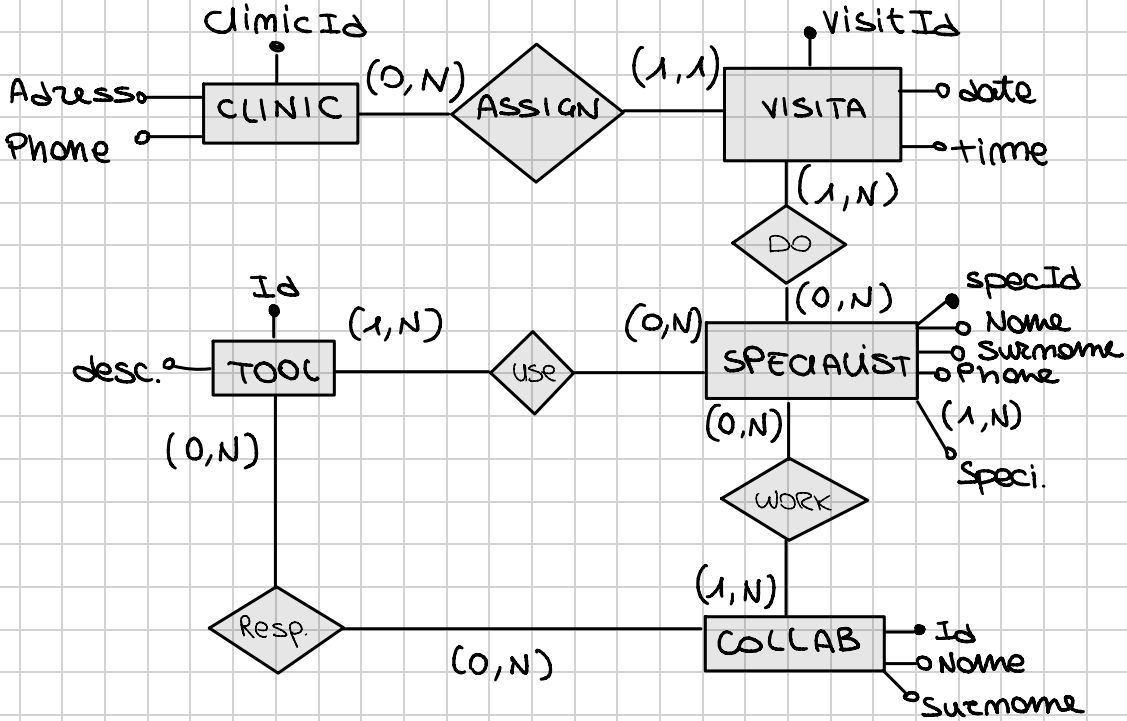
VISITA

SPECIALISTA

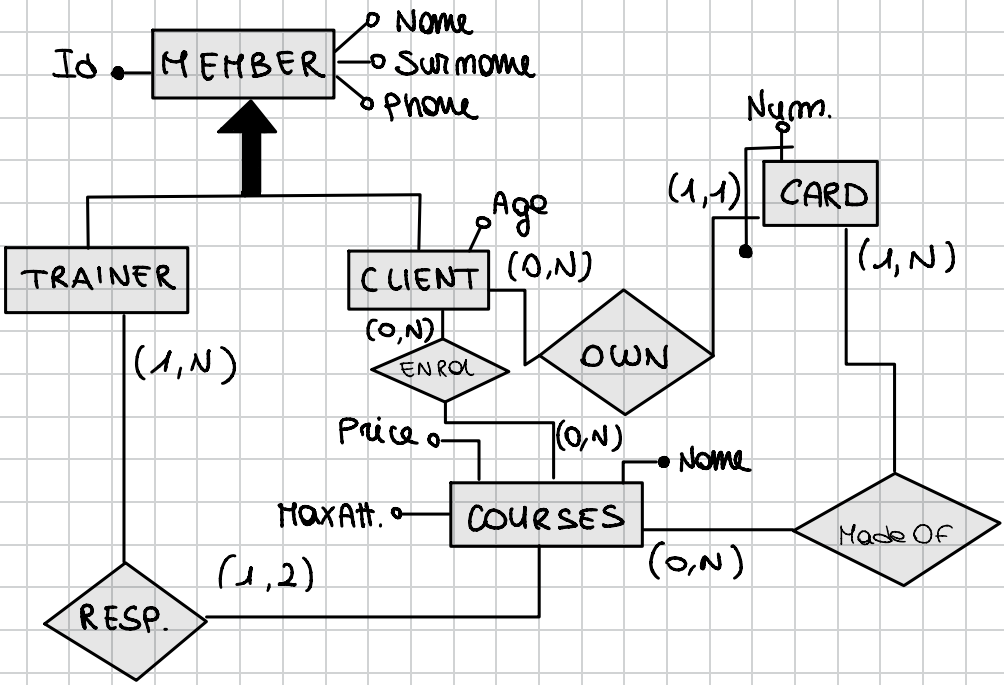
CLINICA

COLLABORATORE

STRUMENTO



ES. 2



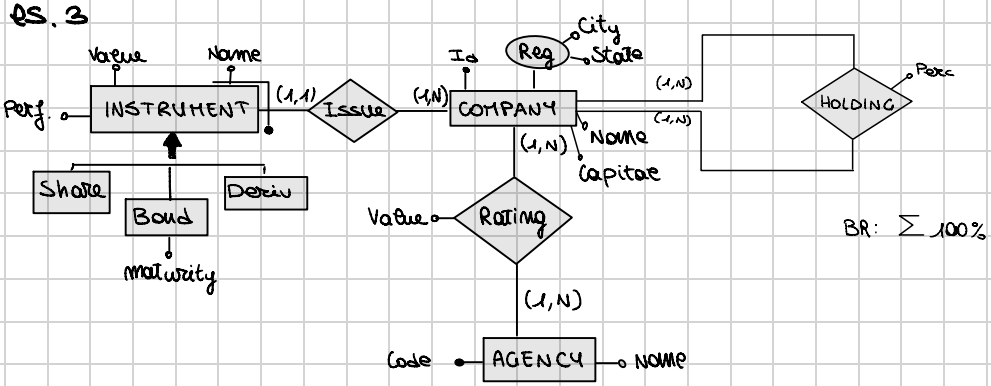
BR0 progressive number

BR1
TOT = Σ Price

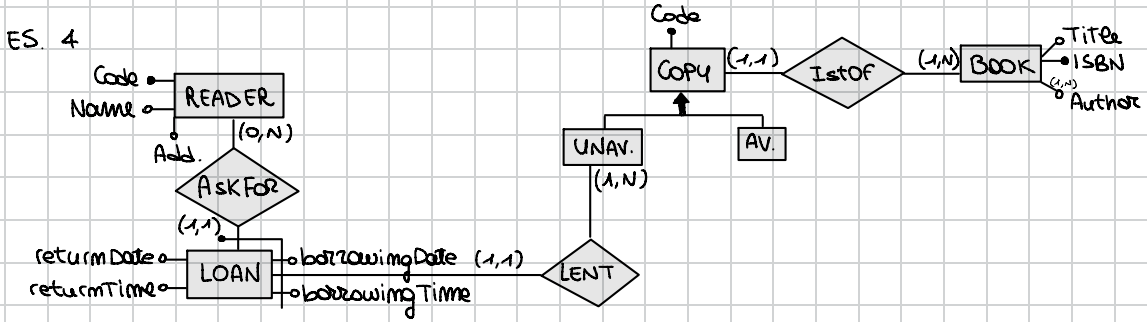
BR2
MaxAtt. $\geq \Sigma$ Clients

27/11/24

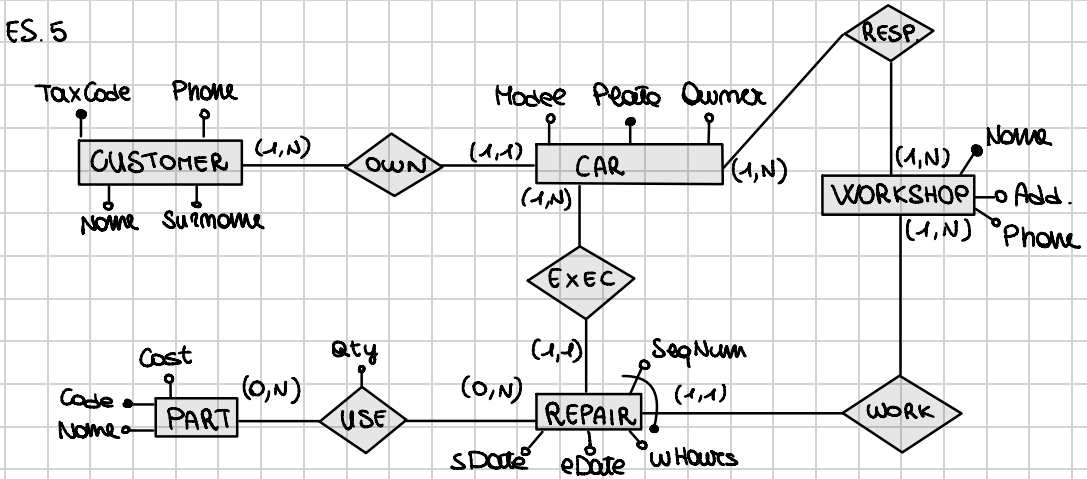
ES. 3



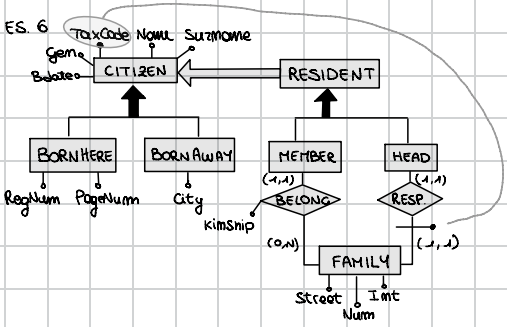
ES. 4



ES. 5



ES. 6



ES. 7

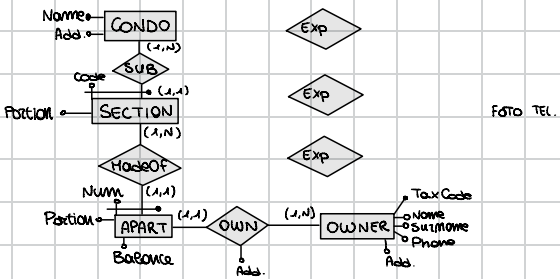
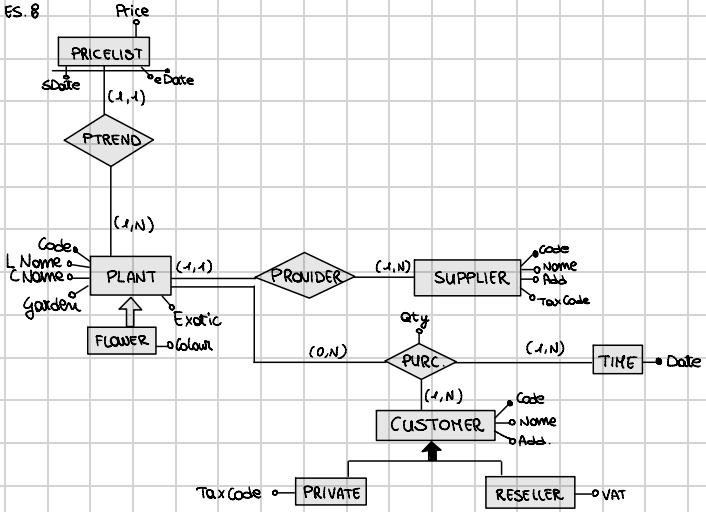


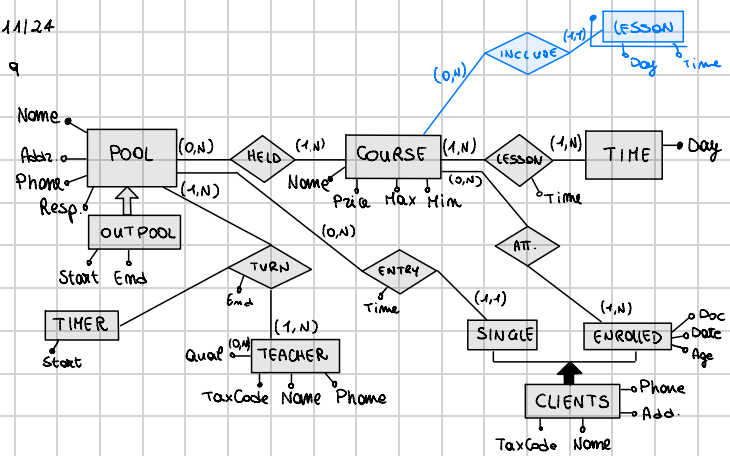
FOTO TEL.

ES. 8

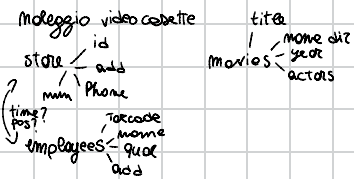


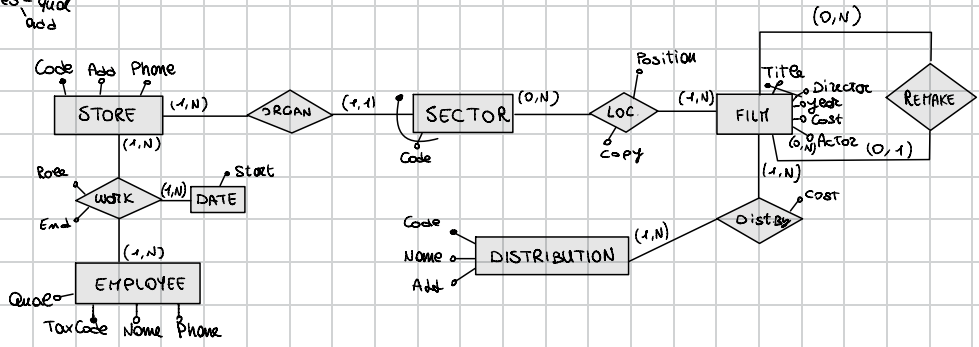
29/11/24

ES. 9

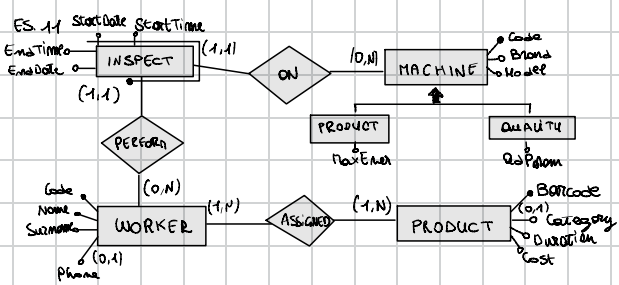


ES. 10

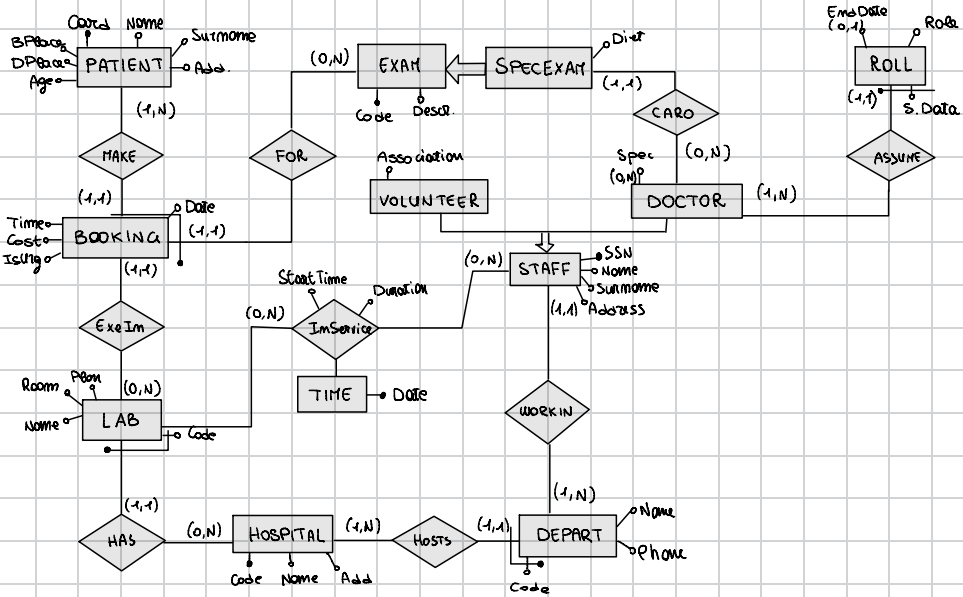




ES. 11



ES. 12

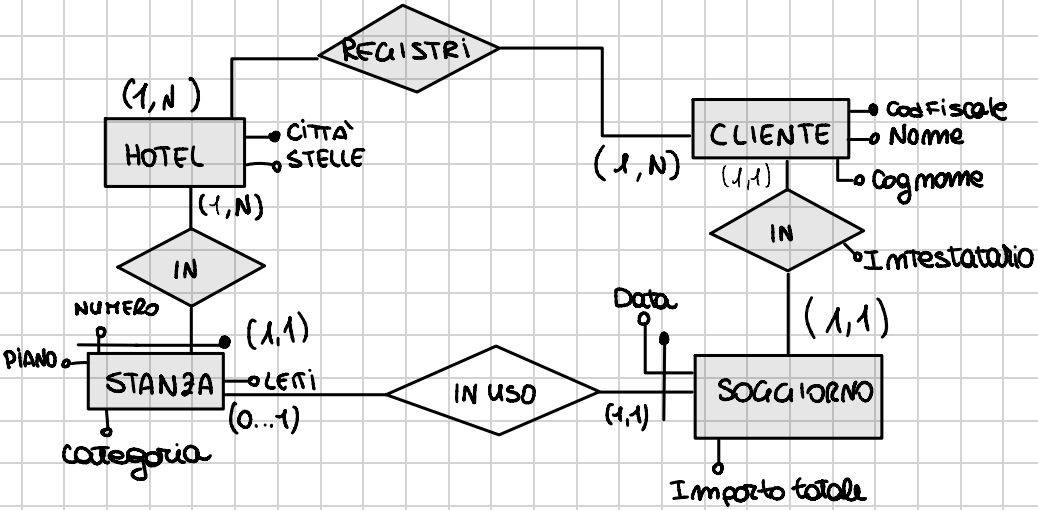

Card
Nome
Surnome
BPlace
DPlace
Age
PATIENT
Add.
(1,N)
MAKE
(1,1)
Date
Time
Cost
Isling
BOOKING
(1,1)
(1,1)
ExeIm
(0,N)
Room
Plan
Nome
LAB
Code
(1,1)
HAS
(0,N)
HOSPITAL
(1,N)
Code
Name
Add
(0,N)
EXAM
Code
Descr.
FOR
Association
VOLUNTEER
Dirr
SPECEXAM
(1,1)
CARO
Spec
(0,N)
(0,N)
DOCTOR
(1,N)
StartTime
Duration
(0,N)
ImService
TIME
Date
(0,N)
STAFF
SSN
Nome
Surnome
Address
(1,1)
WORK IN
(1,N)
HOSTS
(1,1)
DEPART
Name
Phone
Code
EndDate
(0,1)
Role
ROLL
(1,1)
S. Date
ASSUME

TRAINING TEST 1

3) PROGETTAZIONE

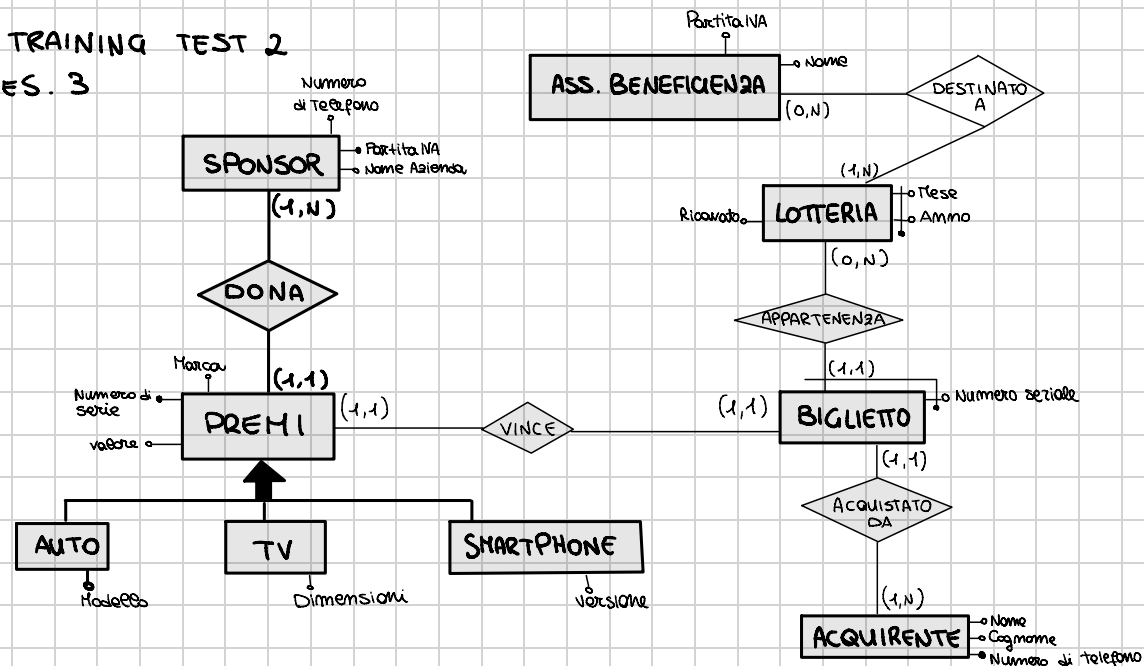
B.R.

$1 < \#\text{LETTI} \leq 3$



TRAINING TEST 2

ES. 3



B.R.: i ricavati sono uguali al prodotto tra il numero di biglietti associati alla lotteria e una costante equivalente al loro prezzo

T.T. 1

ES 4)

1	2
read(x)	
	write(x)
	commit
write(x)	
commit	

=> lost update

l'aggiornamento t_2 è perso dalla sovrascrittura di t_1

1	2
read(x)	
write(x)	
	read(x)
	commit
abort	

=> write-read conflict

t_2 legge un dato potenzialmente invalido

1	2
read(x)	
	read(x)
write(x)	
read(y)	
commit	commit

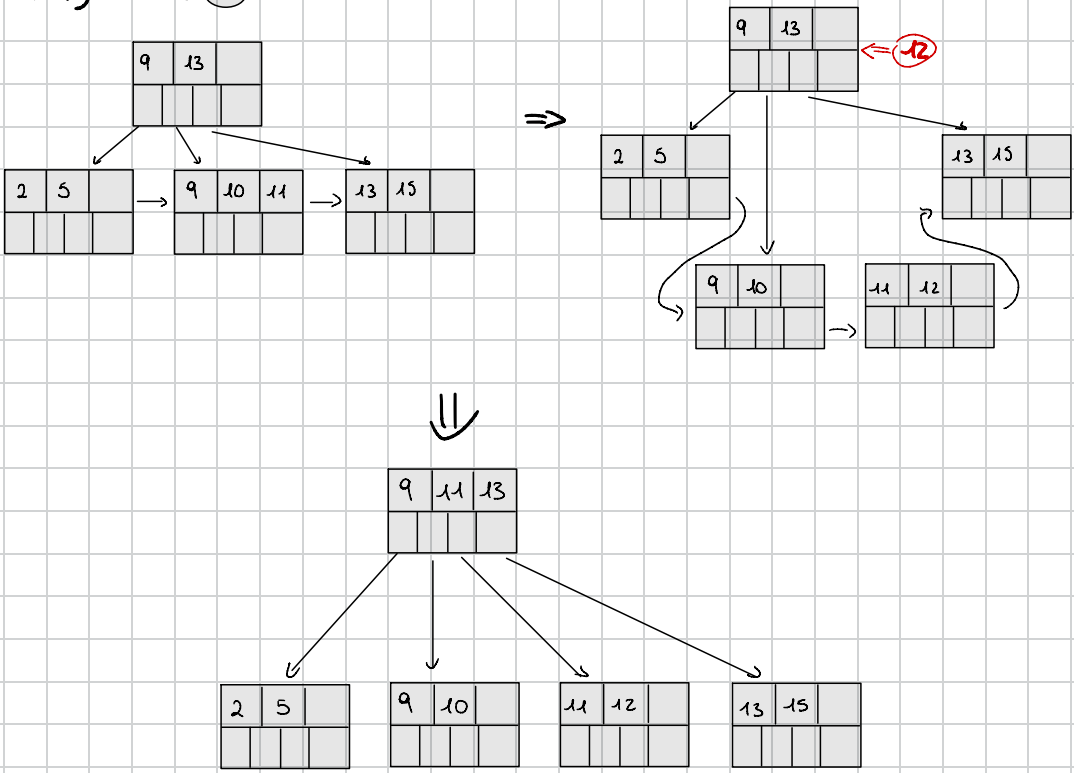
=> ok!

1	2
read(x)	
write(x)	
	read(x)
	write(x)
commit	
	commit

=> ok!

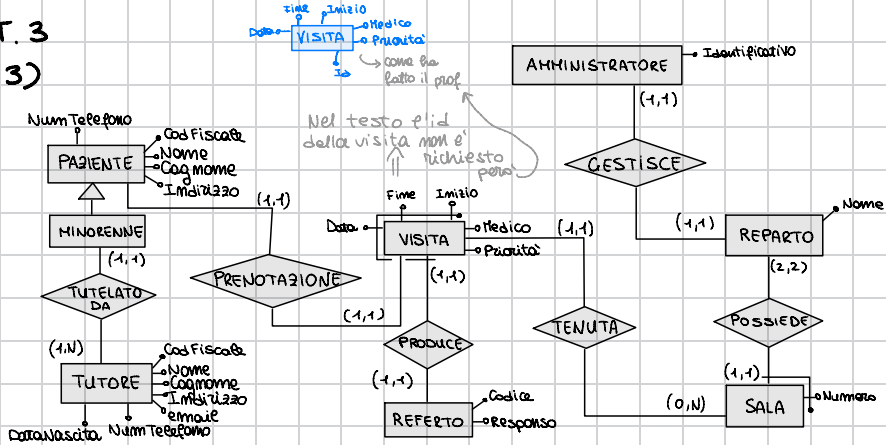
T.T. 2

es. 9) inserire (12)



T.T. 3

es 3)



es 4)

1	2	
write(x)		
	read(x)	dirty reading
	write(y)	inserire lock tra write(x)
abort		e read(x) => se t_1 non e'
	commit	commit non eseguire t_2

1	2
read(x)	
	read(x)
	write(x)
read(y)	
commit	
	commit

=> OK!

con anomalie

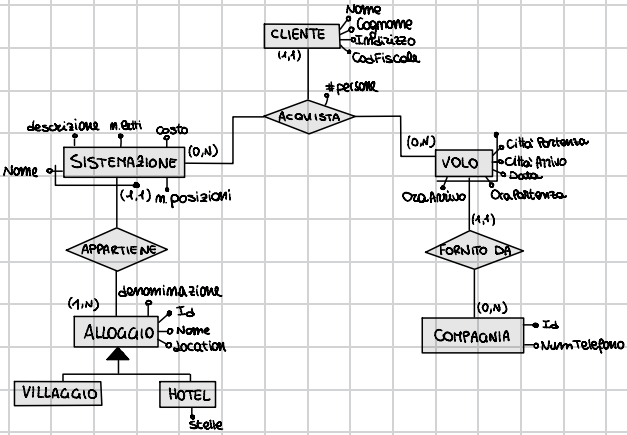
1	2
read(x)	
write(x)	
	read(x)
	write(x)
abort	
	commit

LOCK → prima controllo che t_1 finisca con successo

=> dirty reading: t_2 legge un dato potenzialmente invalido

T.T.4

es 3)


CLIENTE
Nome
Cognome
Indirizzo
CodFiscale
(1,1)
#persone
ACQUISTA
(0,N)
descrizione
m.letti
costo
SISTEMAZIONE
Nome
(1,1)
m.posizioni
APPARTIENE
(1,N)
denominazione
ALLOGGIO
Id
Nome
location
VILLAGGIO
HOTEL
Stelle
(0,N)
VOLO
Citta' Partenza
Citta' Arrivo
Data
Ora Arrivo
Ora Partenza
(1,1)
FORNITO DA
(0,N)
COMPAGNIA
Id
NumTelefono

T.T. 1

es 4)

1.

1	2
read(x)	
	write(x)
	commit
write(x)	
commit	

lost update (write-write)

2.

1	2
read(x)	
write(x)	
	read(x)
	commit
abort	

dirty reading (write-read)

3.

1	2
read(x)	
	read(x)
	write(x)
read(y)	
commit	
	commit

=) OK!

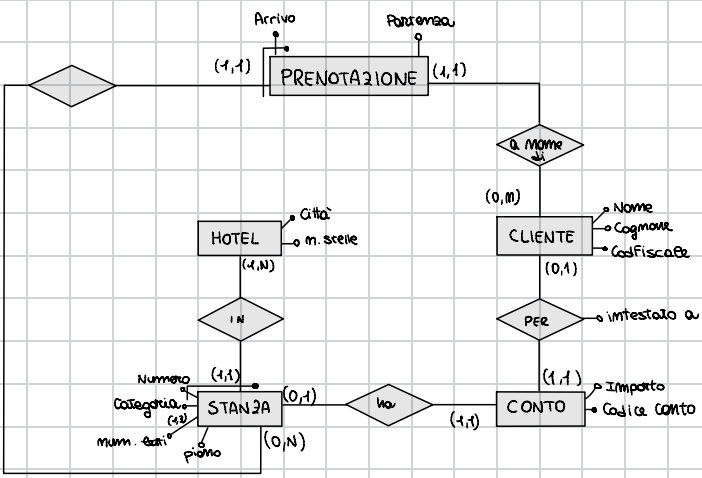
4.

1	2
read(x)	
write(x)	
	read(x)
	write(x)
commit	
	commit

=) OK!

T.T.1

es.3) (di nuovo!)



We want to model a system for the management of a gym and its members.

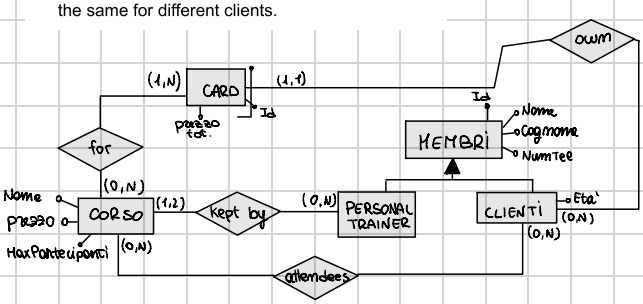
- Each course has a unique name and a fixed price.
- We want to store name, surname and the phone number for all members involved in the gym (both trainers and members).
- Each member has an unique id.
- We want to know the clients' age to fit the exercises workload.
- Each course is taught by one or two trainers.
- Each course has a maximum number of attendees.
- Each member can buy one or more cards, and each card unlocks some courses that are thus purchased at a global price. The final price for each card must not exceed the sum of the prices of the included courses.
- Each client can enrol in a course without buying a card.
- Each card is identified by a progressive number, that is unique for each client. The number can be the same for different clients.

entità:
corso

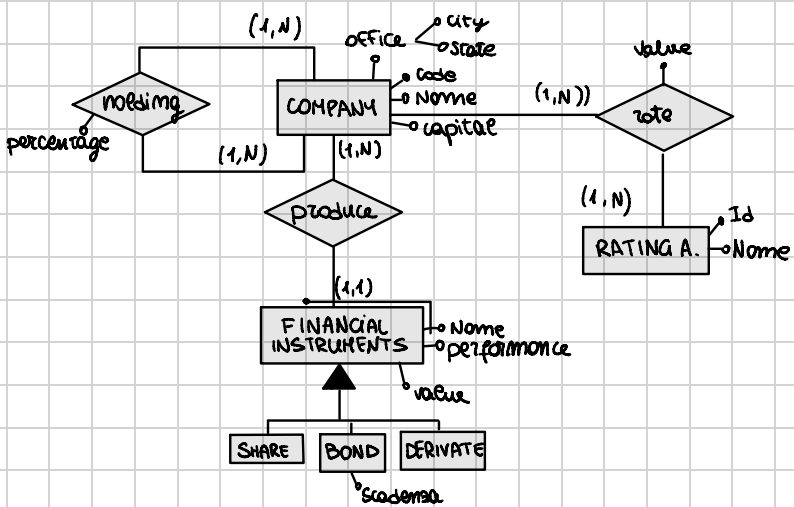
membri (pt e clienti)

pt - corso

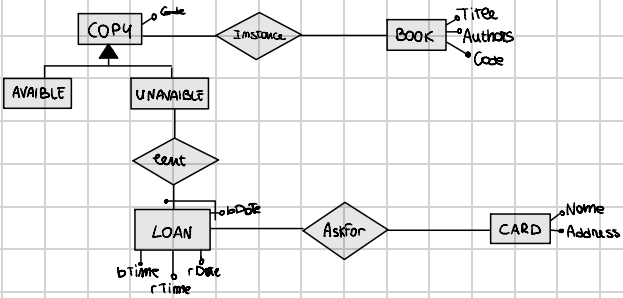
card



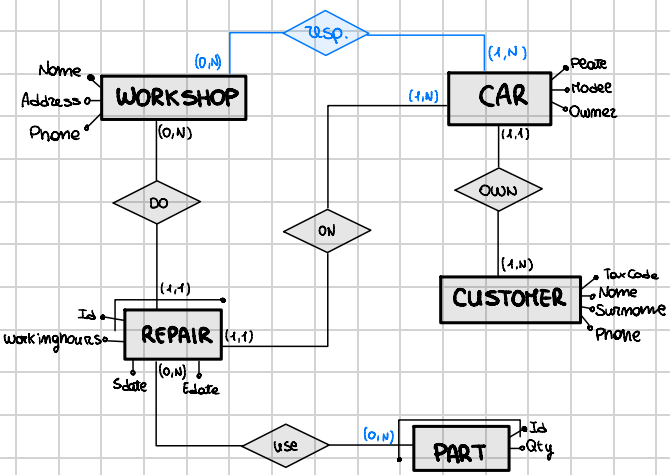
p.33 es.3



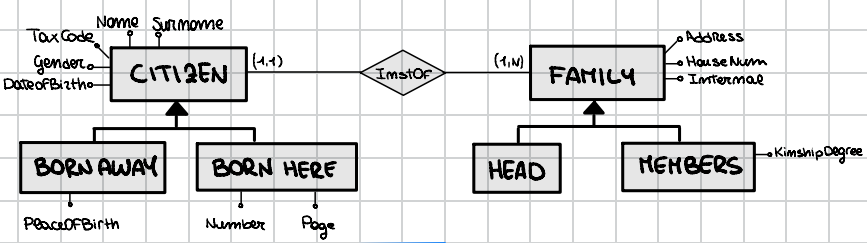
es. 4 p.47

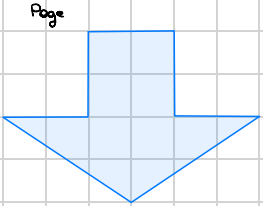


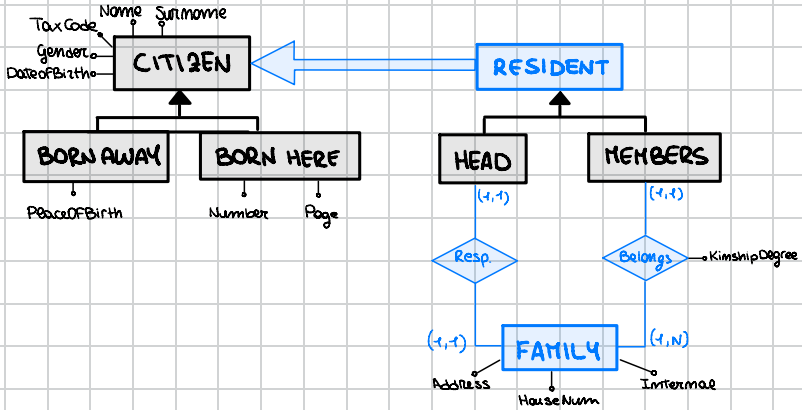
es. 5 p. 50



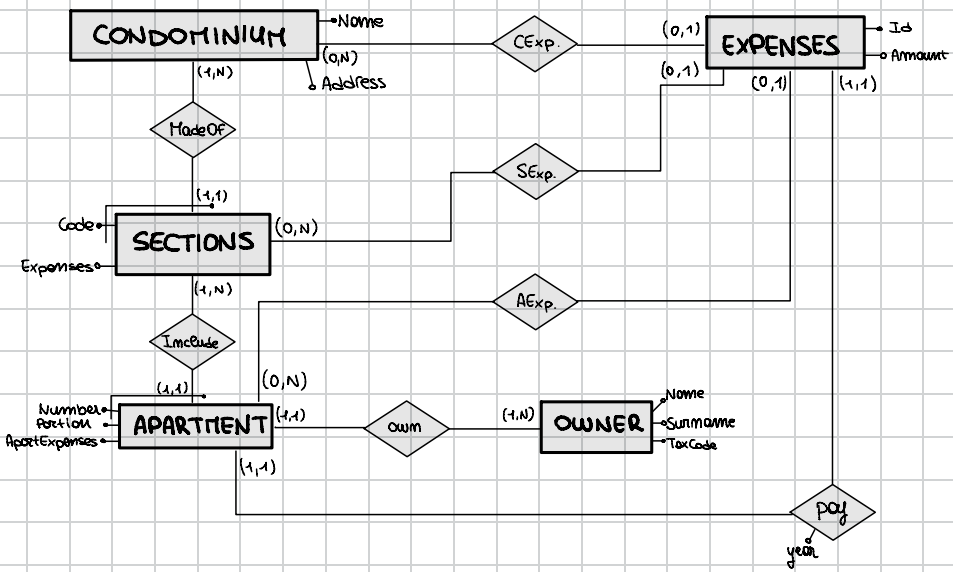
ES. 6 p. 52



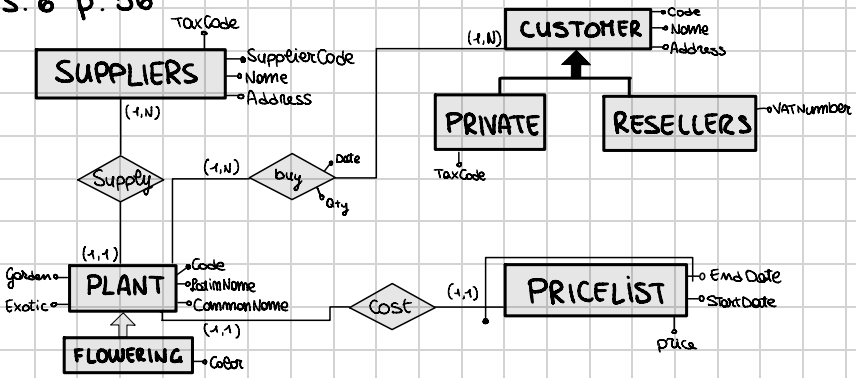



TaxCode
Name
Surname
Gender
DateOfBirth
CITIZEN
RESIDENT
BORN AWAY
BORN HERE
HEAD
MEMBERS
PeaceOfBirth
Number
Page
(1,1)
(1,1)
Resp.
Belongs
KinshipDegree
(1,1)
FAMILY
(1,N)
Address
HouseNum
InternNum

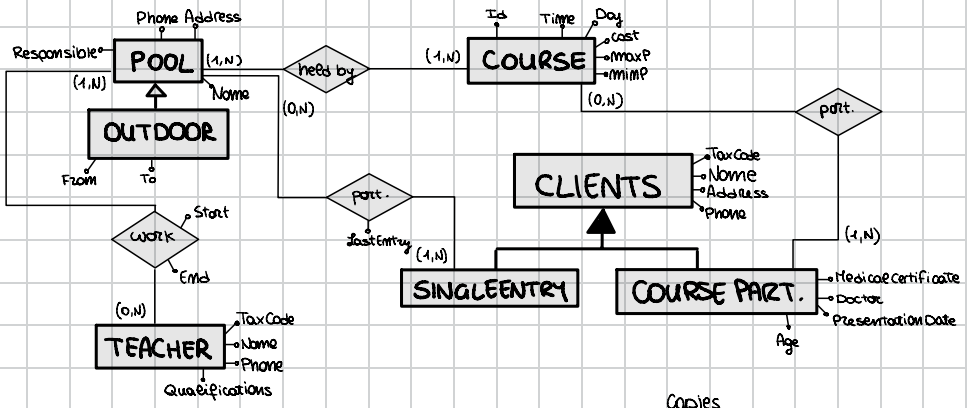
es. 7 p. 54


CONDOMINIUM
Name
(0,N)
Address
CExp.
(0,1)
EXPENSES
Id
Amount
(1,N)
MadeOf
(0,1)
(0,1)
(1,1)
SExp.
(1,1)
Code
SECTIONS
(0,N)
Expenses
(1,N)
AExp.
Include
(1,1)
(0,N)
Number
Portion
ApartExpenses
APARTMENT
(1,1)
own
(1,N)
OWNER
Name
Surname
TaxCode
(1,1)
pay
year

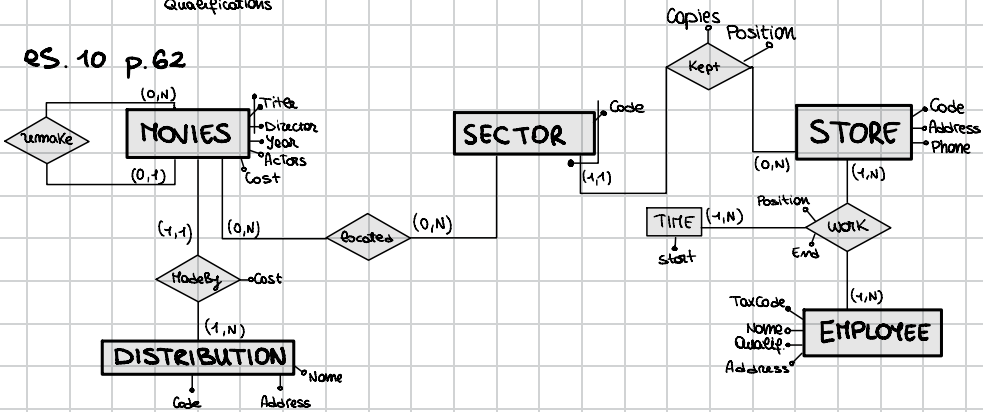
es. 8 p. 56



es. 9 p. 58



es. 10 p. 62



SIMULAZIONE