

G

D



D

da cassetta del masochista

insieme di dati che descrivono una realtà e che voglio interrogare
Database: an organized set of data that supports or carries 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 => LINGUAGGIO
- sistemi x gestire i dati
- metodologie x realizzare / rappresentare una base di dati

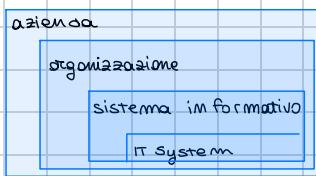
INFORMATION SYSTEM: - componente di un'istituzione che gestisce informazioni

- 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 del sistema informativo (utilizza tecnologie) automatizzata



GESTIONE INFORMAZIONI:

raffresentate 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 interpreto

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

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

DATABASE MANAGEMENT SYSTEM:

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

struttura che consente di gestire le basi di dati (es. equivalente del compilatore)

mentre applica: - privacy - affidabilità - efficienza - efficiacia

- da 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 cp. sullo stesso dato)

- inconsistenza

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

più utenti hanno accesso: la base deve rimanere consistenti (es. male = -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 riconosciuti:

soldi da A e versati su B o nessuna delle due op. viene eseguita."

CONCORRENZA: su un database lavoriamo + persone. PROB: le modifiche fatte da tutte persone

ESERCIZI 2/10/24

es. 1

Region(Name, Surface)

City(Code, Name, Population)

Belong(Region, City)

• $\forall_{region} \left(BELONG \bowtie_{city=code} \sigma_{Population > 1M} (City) \right) \bowtie_{region=name} Region$

• $\forall_{mome} \left(P \left(Region \leftarrow name \left(\text{or}_{surface > 2.5K} (REGION) \right) \right) \bowtie BELONG \bowtie P_{city \leftarrow code} (City) \right)$

es. 2

TRAIN(Code, Start, End, Miles)

$\wedge_{Start = 'Boston' \wedge End = 'Chicago'}$

$P \left(code, switch, miles \leftarrow code, end, miles (TRAIN) \right)$



$P \left(code, switch, miles \leftarrow code, start, miles (TRAIN) \right)$

oppure

$P \left(code, switch, miles \leftarrow code, end, miles (TRAIN) \right)$



$switch = switch \wedge start = 'Boston' \wedge end = 'Chicago'$

$P \left(code, switch, miles \leftarrow code, start, miles (TRAIN) \right)$

oppure

$P(\text{Code}, \text{Switch}, \text{Miles} \leftarrow \text{Code}, \text{End}, \text{Miles} \leftarrow \text{start} = 'Boston', (\text{TRAIN}))$

$\bowtie_{\text{Switch} : \text{Switch}}$

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

es. 3

PERSON (Id, Name, Surname, Age)

ENROL (Person, Course)

COURSE (Name, Price)

$\forall \text{Name} (\sigma_{\text{Price} > \text{Age}} \circ P_{\text{person} \leftarrow \text{Id}} (\text{PERSON}) \bowtie_{\text{ENROL}} P_{\text{course} \leftarrow \text{Name}} (\text{COURSE}))$

09/10/24

SAILOR (Id, Name, Levee, BirthD)

BOOKING (sailor, Boat, Date)

BOAT (Id, Name, Colour)

Maria e Gianni hanno prenotato una barca verde e una rossa.

$\forall \text{Id}, \text{Name} (\text{Sailor} \bowtie_{\text{Id} = \text{sailor}} (\exists_{\text{sailor}} (\text{BOOKING} \bowtie_{\substack{\text{boat} \\ \text{id}}} \text{color} = 'red')) \bowtie (\exists_{\text{sailor}} (\text{BOOKING} \bowtie_{\substack{\text{boat} \\ \text{id}}} \text{color} = 'green'))))$

PLANE (Id, Name, Range)

CERTIF (EId, PId)

EMPLOYEE (Id, Name, Wage)

Id e Nome di piloti che possono volare su almeno 2 aereoplani che possono far 5 mila miglia.

$\forall \text{Id}, \text{Nome} (\text{EMPLOYEE}) \bowtie_{\text{Id} = \text{EId}} ((\exists_{\text{PId}, \text{EId}} (\text{CERTIF} \bowtie_{\text{PId} = \text{Id}} \forall \text{Id} (\sigma_{\text{Range} \geq 5000} (\text{PLANE})))) \bowtie_{\text{EId} = \text{EId} \wedge \text{PId} \neq \text{PId}} (\exists_{\text{PId}, \text{EId}} (\text{CERTIF} \bowtie_{\text{PId} = \text{Id}} \forall \text{Id} (\sigma_{\text{Range} \geq 5000} (\text{PLANE}))))))$

SHOW (Code, Title, Comp, Duration)

PLAYBILL (Date, Time, Show, Theatre)

Comp che hanno fatto uno spettacolo il 15/10 alle ore 16

$\exists \text{comp} (\exists \text{id_comp} (\text{SHOW } \Delta \text{code} = \text{show} \wedge \exists \text{Date} = \text{15/10} (\text{PLAYBILL})) -$

$\exists \text{id_Comp} (\text{SHOW } \Delta \text{code} = \text{show} \wedge \exists \text{15/10} (\text{PLAYBILL}))$

■ EMPLOYEE (Code, Surname, Wage, Dept.)

DEPARTMENT (Id, Name, Location, Director)

PROJECT (Num, Name, Budget, Manager)

STAFF (Employee, Project)

Name imp. che lavorano a fij > 100k

$\exists \text{surname} (\text{EMPLOYEE} \Delta \text{Project} (\exists \text{code} = \text{employee} \wedge \exists \text{budget} > 100k \text{ PROJECT}))$

Surname impiegati che guadagnano + del manager

$\exists \text{surname} (\text{EMPLOYEE} \Delta \text{DEPARTMENT} \Delta \text{Project} (\exists \text{Dept} = \text{Id} \wedge \exists \text{wage} > \text{w} \wedge \exists \text{director} = \text{c} \wedge \exists \text{code}, \text{surname}, \text{wage}, \text{Dept} \in \text{Employee}))$

Surname capi dept dei project manager

$\exists \text{surname} ((\text{EMPLOYEE} \Delta \text{DEPARTMENT}) \Delta \text{code} = \text{Director} \wedge \text{U})$

$\exists \text{surname} ((\text{EMPLOYEE} \Delta \text{PROJECT}) \Delta \text{code} = \text{manager})$

- Nome dept dove wage.employee > 60k

$\exists \text{Name} (\exists \text{Dept} = \text{Id} (\text{DEPARTMENT} \Delta \text{EMPLOYEE}) \Delta \text{code} = \text{Dept} \wedge \exists \text{Dept} = \text{Id} (\text{EMPLOYEE} \Delta \text{DEPT}))$

$\exists \text{surname} ((\text{EMPLOYEE} \Delta \text{DEPARTMENT}) \Delta \text{Dept} = \text{Id} \wedge \text{code} = \text{Director})$

cognome impiegati con lo stipendi + alto

$\exists \text{code}, \text{surname} ((\text{EMPLOYEE} \Delta \text{EMPLOYEE}) \Delta \text{wage} < \text{w} \wedge \exists \text{code}, \text{surname}, \text{wage}, \text{Dept} \in \text{Employee})$

■ SAILOR (Id, Name, Level, BirthD)

BOOKING (Sailor, Boat, Date)

BOAT (Id, Name, color)

- Maximai che hanno prenotato sia una barca verde che una rossa

$\forall \text{Id}, \text{Name} \in \text{SAILOR}$ $\exists \text{Id} = \text{sailor}$

 $(\exists \text{Sailor}, \text{Boat} \in \text{BOOKING} \wedge \text{Boat} = \text{Id} \wedge (\forall \text{Id}, \text{color} \in \text{BOAT} \text{ s.t. } \text{color} = \text{'RED'}))$

\wedge

 $(\exists \text{Sailor}, \text{Boat} \in \text{BOOKING} \wedge \text{Boat} = \text{Id} \wedge (\forall \text{Id}, \text{color} \in \text{BOAT} \text{ s.t. } \text{color} = \text{'GREEN'}))$

| |
|--|
| SAILOR |
| <u><u>Id</u></u> Name Level BirthD |

\wedge

| |
|----------------------|
| BOOKING |
| Sailor Boat Date |

$\times \text{Id} = \text{sailor}$

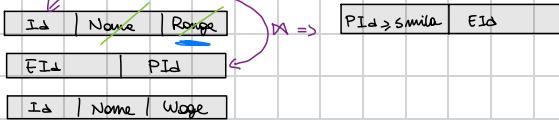
| |
|-----------------------|
| Sailor Boat Color |
|-----------------------|

| |
|---------------------------------|
| BOAT |
| <u><u>Id</u></u> Name Color |

■ 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 ≥ mila miglia

$\forall \text{Id}, \text{Name} \in \text{EMPLOYEE}$ $\exists \text{Id} = \text{EId}$

 $(\exists \text{PId}_1, \text{EId}_1 \in \text{CERTIF} \wedge \exists \text{PId}_2, \text{EId}_2 \in \text{CERTIF} \text{ s.t. } \text{PId}_1 = \text{Id} \wedge \text{PId}_2 = \text{Id} \wedge (\exists \text{Range}_{\geq 5000} \in \text{PLANE} \text{ s.t. } \text{Range} = \text{Range}_{\geq 5000}))$

\wedge

$\exists \text{EId}_1 = \text{Id} \wedge \exists \text{EId}_2 = \text{Id}$

 $(\exists \text{EId}_1 = \text{Id} \wedge \exists \text{EId}_2 = \text{Id} \wedge (\exists \text{Range}_{\geq 5000} \in \text{PLANE} \text{ s.t. } \text{Range} = \text{Range}_{\geq 5000}))$

■ SHOW(Code, Title, Comp, Duration)

| | | | |
|------|-------|------|----------|
| Code | Title | Comp | Duration |
|------|-------|------|----------|

PLAYBILL (Date, Time, Show, Theatre)

| | | | |
|------|------|------|---------|
| Date | Time | Show | Theatre |
|------|------|------|---------|

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

$\forall \text{Id}, \text{Comp}$

 $\exists \text{comp} \in \text{SHOW}$ $\exists \text{Code} = \text{Show}$ $(\exists \text{Date} = 15/10 \text{ in } \text{PLAYBILL})$

$\forall \text{Id}, \text{Comp}$

 $\forall \text{SHOW}$ $\exists \text{Code} = \text{Show}$ $(\exists \text{Date} = 16/10 \text{ in } \text{PLAYBILL})$

ES. 2.8

■ 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

\forall code, surname

$((\exists \text{employee} (\text{P}_{\text{E}, \text{P}} \leftarrow \text{Employee, Project}) \wedge \text{STAFF})) \rightarrow \exists \text{Employee} = \text{code} (\forall \text{code, surname} (\text{EMPLOYEE}))$

■ 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

$\forall \text{code, surname} (((\exists \text{employee} (\text{STAFF})) - (\exists \text{employee} (\text{P}_{\text{E}, \text{P}} \leftarrow \text{Employee, Project}) \wedge \text{STAFF})) \rightarrow \exists \text{Employee} = \text{code} (\forall \text{code, surname} (\text{EMPLOYEE})))$

ES. LIBRO

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

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

INTERPRETAZIONI(CodiceFilm, CodiceAttore, Personaggio)

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

$\forall \text{CodiceFilm, CodiceAttore, Personaggio, Cognome, Nome, Sesso, DataNascita, Nazionalita, Titolo, Regista, Ammo, CostoNoleggio}$

 $((((P_{\text{CodiceFilm} \leftarrow \text{CodiceFilm}} (\text{FILM})) \wedge \text{CodiceFilm} = \text{CodiceFilm}) \wedge \text{INTERPRETAZIONI}))$

\wedge

 $((P_{\text{CodiceAttore} \leftarrow \text{codiceAttore}} (\text{ARTISTI})) \wedge \text{CodiceAttore} = \text{CodiceAttore}) \wedge \text{INTERPRETAZIONI})$

• titoli film interpretati da Henry Fonda

$\forall \text{Titolo}$

$((\forall \text{CodiceAttore, CodiceFilm} (((P_{\text{CodiceAttore} \leftarrow \text{CodA}} (\forall \text{CodiceAttore} (\Theta \text{Nome} = \text{Henry}, (\text{ARTISTI}))) \wedge \text{Cognome} = \text{'Fonda'})) \wedge \text{CodA} = \text{Codice Attore}) \wedge \text{CodiceFilm} = (P_{\text{CodiceFilm} \leftarrow \text{CodF}} (\forall \text{CodiceFilm} (\text{INTERPRETAZIONI})))) \wedge \text{CodiceFilm} = (P_{\text{CodiceFilm} \leftarrow \text{CodF}} (\forall \text{CodiceFilm} ((\forall \text{Titolo} (\text{FILM}))))))$

$\wedge \text{CodA} = \text{Codice Attore}$

$(\forall \text{CodiceFilm, CodiceAttore})$

$(\forall \text{CodiceFilm} = (P_{\text{CodiceFilm} \leftarrow \text{CodF}} (\forall \text{CodiceFilm} ((\forall \text{Titolo} (\text{FILM}))))))$

• titoli dei film per i quali il regista è stato anche interprete

$$\pi_{\text{titolo}} ((\pi_{\text{Regista}, \text{CodiceFilm}, \text{titolo}} (\text{FILM})) \bowtie_{\text{Regista} = \text{cognome}} (\pi_{\text{codiceAttore}, \text{cognome}} (\text{ARTISTI})))$$

• titolo dei film in cui gli attori mai siamo tutti dello stesso sesso. $H = M \wedge F = M$

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

| | | | |
|-------|--------------------------|------------------------|------------------------|
| Cod A | Codice Attore | Codice Film | Personaggio |
|-------|--------------------------|------------------------|------------------------|

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

$$\pi_{\text{Codice Film}} (\text{FilmM} \cap \text{FilmF})$$

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

} meglio evitare le query annidate

INTERSECT

SELECT Surname

FROM STUDENT)

• SELECT DISTINCT Surname

FROM LECTURER, STUDENT

WHERE LECTURER.Surname =

• SELECT DISTINCT Surname

FROM LECTURER JOIN STUDENT

ON Surname = Surname

STUDENT.Surname

• Q: Cognome dei docenti che fanno 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, Alias, Age)

FILM (FilmId, Title, Year, Director)

Q: titolo film prodotto 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)

Album

Playlist con prima del 2001 appartenenti UMG

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 = 'UMG'

Playlist con canzoni pubblicate prima del 2001 di artisti appartenenti a 'UMG'.

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 = 'Federico' 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 il mese 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 = '2023-12-23'

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.Name, 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.Name
```

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 6 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 pro. e incasso a Roma che a gennaio 2005 hanno guadagnato 20 mila €.

```
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 < '31/01/2005'
```

```
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 EXISTS (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 EXISTS  
                      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(Name, City)

ARTIST(Name, Nation)

WORK(Code, Title, NameM, NameA)

CHARACTER(Name, CodeW)

Nome Musei di Londra che non hanno opere di Tiziano

```
SELECT M.Nome  
FROM MUSEUM AS M  
WHERE M.City = 'London' AND NOT EXISTS ( SELECT *
```

```
    FROM WORK AS W  
    WHERE W.NomeA = 'Tiziano' AND M.Nome = W.NomeM )
```

Q: Nomi musei a Londra che fanno 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, CodeJms)

OWNERS (Code, Name, Residence)

INSURANCE (CodeJms, Name, Hq)

ACCIDENT (CodeAcc, Location, Date)

INVCAR (CodeAcc, Plate, Amount)

• AUTO BREV 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.CodeJms = I.CodeJms
```

```
WHERE ( C.Brand = 'BMW' OR C.Power > 120 ) AND I.Name = 'AVIVA'
```

Q: ogni assicurazione nome ✓ e #auto assicurate

```
SELECT I.Name, I.Hq, COUNT(*)
```

```
FROM INSURANCE AS I JOIN CARS AS C
```

```
ON I.CodeJms = C.CodeJms
```

```
GROUP BY I.Name, I.Hq, I.CodeJms
```

Q: Auto coinvolte in un incidente e sostituisco targa, nome, dommi totale.

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

SIMULAZIONE D'ESAME

MODELLO (IdModello, IdMarca, Nome, Categoria)

VALUTAZIONE (IdModello, Ammo, Prezzo)

MARCA (IdMarca, Motocchio, Nazionalita')

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

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

Q: togliere le valutazioni delle berline prodotte dalle case francesi che non fanno valutazione per l'anno 2007.

$\neg \exists IdModello, Prezzo, Ammo$
 $(\theta \text{ Nazionalita}' = \text{'Francia'} \wedge ((\text{MARCA} \bowtie \text{MODELLO}) \bowtie (\text{VALUTAZIONE} - (\theta \text{ Ammo} = 2007 \wedge (\text{VALUTAZIONE}) \wedge \text{Prezzo} \neq \text{NULL}))))$
 $\wedge \text{Categoria} = \text{'Berlina'}$

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

$\neg \exists IdModello, prezzo, anno$
 $(\theta \text{ Prezzo} > 15000 \text{ AND } 2005 < \text{Anno} < 2009 \text{ AND } \text{Categoria} = \text{'utilitaria'} \wedge \text{Nazionalita}' \neq \text{'Italia'}) \bowtie (\text{MARCA} \bowtie \text{MODELLO} \bowtie \text{VALUTAZIONE})$

ES. 3.17

- $\exists_{\text{NomeFarmaco}, \text{Nome}, \text{Codice}} (\text{FARMACI} \bowtie \text{PRODUTTORI} \bowtie (\theta_{\text{categria}=\text{"suefarmatica"}, \text{SOSTANZE}}))$
- $\exists_{\text{NomeFarmaco}, \text{NomeSostanza}, \text{Codice}} (\text{FARMACI} \bowtie (\theta_{\text{nazione}=\text{"Italia"}}, \text{PRODUTTORI}) \bowtie (\text{SOSTANZE}))$

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)

10:

$\exists_{\text{nameP}, \text{ProfessorId}}$

$(\text{P}_{\text{Dep}, \text{NameP}} (\text{PROFESSOR} \bowtie \text{TEACHES} \bowtie \text{ENROLLMENT} \bowtie \text{MAJOR})$
 $\quad \leftarrow \text{Department, Name}$

$\bowtie_{\text{Dep}=\text{Department}} (\text{STUDENT} \bowtie \text{MAJOR})$)

SANJO:

$\exists_{\text{name}, \text{ProfessorId}}$

$((\text{PROFESSOR} \bowtie \text{TEACHES} \bowtie \text{ENROLLMENT} \bowtie \text{MAJOR})$

$\bowtie_{\text{department}} (\text{STUDENT} \bowtie \text{MAJOR})$)

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

$\exists_{\text{StudentId}} (\text{STUDENT} \bowtie \text{ENROLLMENT}$

$\bowtie (\theta_{\text{Department}=\text{"Computer Science"}}, (\text{COURSE}))$

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)

\uparrow
 \exists StudentId, Name

$((\exists \text{ ProfessorId} (\Theta \text{ Department} = \text{'History'}) (\text{PROFESSOR})))$

\bowtie
 \exists ProfessorId, CourseId
(TEACHES)

\bowtie

\exists CourseId, StudentId (Θ Grade = 'A' (ENROLLMENT))

\bowtie

$(\exists \text{ StudentId, Name} (\text{STUDENT}))$

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

\uparrow
 \exists StudentId, Name ($((\exists \text{ ProfessorId} (\Theta \text{ Department} = \text{'Physics'}) (\text{PROFESSOR})))$)

\bowtie
 \exists ProfessorId, CourseId
(TEACHES) \bowtie \exists CourseId, StudentId (ENROLLMENT)

\bowtie
 \exists StudentId, Name (STUDENT))

ESERCITAZIONE 28/10 ALGEBRA

ES. 3

$\text{TRAIN}(\underline{\text{Code}}, \text{Start}, \text{End}, \text{Miles})$

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

$P_{\text{Code}, \text{Switch}, \text{Miles}}(\text{TRAIN})$

$\leftarrow \text{Code}, \text{End}, \text{Miles}$ $\wedge \text{Start} = \text{'Boston'} \wedge \text{End} = \text{'Chicago'}$
 $\wedge \text{switch} = \text{switch}$

$P_{\text{Start} \leftarrow \text{switch}}(\text{TRAIN})$

ES. 4

$\text{PERSON}(\underline{\text{Id}}, \text{Nome}, \text{Surname}, \text{Age})$

$\text{ENROLLMENT}(\underline{\text{Person}}, \underline{\text{Course}})$

$\text{COURSE}(\underline{\text{Nome}}, \underline{\text{Price}})$

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

$\forall \text{Nome}$

$(\exists \text{Price} > \text{Age} \cdot 10) (((\forall \text{Id}, \text{Name}, \text{Age} (\text{PERSON}))$

$\wedge \text{Id} = \text{Person}$ ENROLLMENT

$\wedge \text{course} = \text{surname} (\text{COURSE})))$

ES. 5

$\text{STUDENT}(\underline{\text{Id}}, \text{Nome}, \text{Surname}, \text{School})$

$\text{EXAM}(\underline{\text{Student}}, \underline{\text{Subject}}, \text{Mark}, \text{Date})$

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

$\forall \text{Nome}, \text{Surname}$

$(\forall \text{student} (\exists \text{Mark} = 30 (\text{EXAM}))$

$\wedge \text{student} = \text{Id} (\forall \text{Id}, \text{Nome}, \text{Surname} (\text{STUDENT})))$

Q: Nome e cognome degli studenti che non fanno mai peso
30

$\pi_{Id, Name, Surname} (\text{STUDENT})$

$\pi_{Id, Name, Surname} (\text{STUDENT}) \bowtie_{\substack{Id = \text{Student} \\ Grade = 30}} (\theta_{\text{EXAM}})$

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'

$\pi_{Name, Id} ($

$\bowtie_{\substack{\text{ProdId}, \text{SupId}}} ($

$((P \bowtie_{\substack{\text{SupId1}, \text{ProdId1} \\ \leftarrow \text{SupId}, \text{ProdId}}} (\theta_{\substack{\text{ProdId} = \text{PX274}}} (\text{SUPPLYING})))$

$\bowtie_{\substack{\text{SupId1} = \text{SupId} \wedge \text{ProdId1} \neq \text{ProdId}}} (\text{SUPPLYING}))$

$\bowtie_{\substack{\text{SupId} = \text{Id}}} (\text{SUPPLIER})$

TROPPO COMPLICATO ↑



$\forall \text{ nome} \left(\text{SUPPLIER} \wedge \text{Id} = \text{SupID}$
 $\quad \left(\exists \text{ ProdID} \neq 'PX274' \text{ (SUPPLYING)} \right) \right)$

Q: codice dei fornitori che forniscono almeno due prodotti diversi

$\forall \text{ SupID}$
 $\left(\left(\left(\text{P}_S1, \text{P}_1 \in \text{SupID}, \text{ProdID} \right) \text{ (SUPPLYING)} \right)$
 $\quad \wedge \text{ S1-SupID} \wedge \text{ P}_1 \neq \text{ProdID}$
 $\quad \text{SUPPLYING} \right)$

ES.9

BOOK (BookID, Title, Author)

USER (User ID, Name, Surname)

BORROWING (User ID, Book ID, Date)

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

$\forall \text{ User ID} \left(\left(\exists \text{ Author} = 'Fleming' \text{ (BOOK)} \right)$
 $\quad \wedge \text{ BORROWING} \right)$

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

$$\begin{aligned}
 \text{HeadWage} := & \uparrow_{\text{code}, \text{wage}, \text{Id}} \\
 & (\text{DEPARTMENT} \bowtie_{\text{Director} = \text{code}} \text{EMPLOYEE})
 \end{aligned}$$

γ_{surname}

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

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

$$\begin{aligned}
 \text{SurnameManager} := & \\
 & \uparrow_{\text{code}, \text{surname}} (\text{PROJECT} \bowtie_{\text{Manager} = \text{code}} \text{EMPLOYEE})
 \end{aligned}$$

$\gamma_{\text{code}, \text{surname}}$

$$\begin{aligned}
 \text{SurnameDirector} := & \\
 & \uparrow_{\text{code}, \text{surname}} (\text{DEPARTMENT} \bowtie_{\text{Director} = \text{code}} \text{EMPLOYEE}) \\
 & \uparrow_{\text{surname}} (\text{SurnameManager} \cup \text{SurnameDirector})
 \end{aligned}$$

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 60K

$$\begin{aligned}
 & \forall \text{ name} \\
 & \left(\left(\forall \text{ Department, name} \left(P_{\text{Department} = \text{Id}} (\text{DEPARTMENT}) \right) \right) \right. \\
 & \quad - \\
 & \quad \left. \left(\forall \text{ Department, name} \left(\Theta_{\text{wage} < 60K} ((\text{EMPLOYEE}) \bowtie_{\text{Department} = \text{Id}} (\text{DEPARTMENT})) \right) \right) \right)
 \end{aligned}$$

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

$$\begin{aligned}
 & \forall \text{ Code, Surname} (\text{STAFF}) \\
 & \quad - \\
 & \forall \text{ code, surname} (\text{EMPLOYEE} \bowtie_{\text{code} = \text{Employee}} \text{STAFF})
 \end{aligned}$$

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 working on more than one project.

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

$$\uparrow \text{code, surname} (((\rho_{E1, P1 \in \text{Employee, Project}} (\text{STAFF})) \\ \bowtie_{E1 = \text{Employee} \wedge P1 \neq \text{Project}} \text{STAFF}) \bowtie \text{EMPLOYEE})$$

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 the employees working on a single project.

Codice e cognome di impiegati che lavorano su un solo progetto

$$\uparrow \text{code, surname} ((\text{STAFF} - ((\uparrow_{\text{Employee, Project}} (\rho_{E1, P1 \in \text{Employee, Project}} (\text{STAFF}))) \\ \bowtie_{E1 = \text{Employee} \wedge P1 \neq \text{Project}} \text{STAFF}))) \bowtie_{\text{Employee} = \text{Code}} \text{EMPLOYEE})$$

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 surname of the highest paid employee.

$$\pi_{\text{Code}, \text{Surname}} ((\rho_{w \leftarrow \text{Wage}} (\text{EMPLOYEE})) \bowtie_{w > \text{Wage}} \text{EMPLOYEE})$$

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 the employees working on a single project.

$$\pi_{\text{Code}, \text{Surname}} (\pi_{\text{Code}} (\text{EMPLOYEE}))$$

$$(\rho_{\substack{\text{Code} \\ \leftarrow \text{Employee}}} (\pi_{\text{Employee}} (((\rho_{\substack{E, P \in \text{Employee}, P \in \text{Project}}} (\text{STAFF}))$$

$$\bowtie_{\substack{E = \text{Employee} \wedge P \in \text{Project} \\ \text{STAFF}}))))))$$

$$\bowtie \text{EMPLOYEE})$$

Cognome dell'impiegato
più pagato

Codice e cognome impiegati che lavorano ad un solo progetto

ESAME 1

M.1

momi 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) sta' minima valutazione dei utenti che hanno dato a 'Blade Rummel' 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 Rummel'
AND V.VALUTAZIONE > 8
```

m.3) valutazione dei film dal 2000 in poi per l'utente AntonioBionchi

$$\pi_{VALUTAZIONE, CODFILM}((\theta_{ANNO \geq 2000}(FILM)) \bowtie VOTO \bowtie (\theta_{ALIAS='AntonioBionchi'}(UTENTE)))$$

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

$$\pi_{CODUTENTE, ALIAS, ETA'}(UTENTE \bowtie ((\theta_{TITOLO='Django'}(FILM)) \bowtie VOTO - (\theta_{TITOLO='Pulp Fiction'}(FILM)) \bowtie VOTO))$$

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 S1.ID)
```

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

$$\nexists_{\text{NOME}, \cancel{\text{X}}(\text{SERIE})} - \nexists_{\text{NOME}, \text{id}}((P_{\text{id} \leftarrow \text{id}}(\text{SERIE} \bowtie_{\text{id} = \text{idSERIE}} (P_{\text{id} \leftarrow \text{id}}(\text{EPISODIO}))) \bowtie_{\text{id} = \text{idEPISODIO}} (\theta_{\text{LINGUA} = \text{'ITALIANO'}}(\text{SOTTOTITOLO}))))$$

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

$\exists_{IDE}^{NUMERO, STAGIONE, ((\theta_{NOME = 'The Walking Dead'} (SERIE))}$

$\wedge_{ID=IDSERIE}$

$(P_{IDE \leftarrow ID} (\theta_{DURATA > 60} (EPISODIO)))$

$\wedge_{IDE = IDEPISODIO}$

$(\theta_{LINGUA = 'INGLESE'} (SOTTOTITOLO)))$