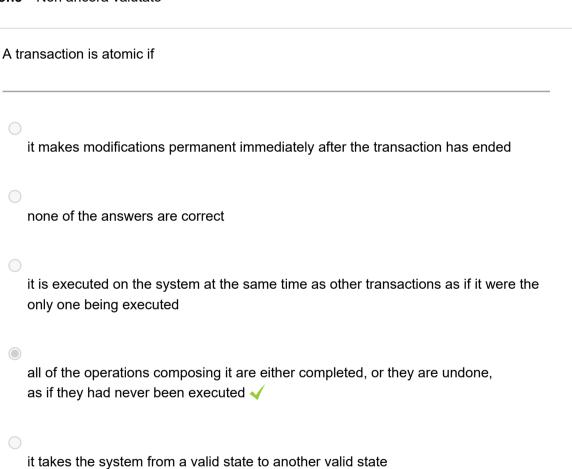
Introduction to databases

Iniziato	martedì, 15 giugno 2021, 09:08
Stato	Completato
Terminato	martedì, 15 giugno 2021, 11:01
Tempo impiegato	1 ora 53 min.
Valutazione	Non ancora valutato

Domanda 1

Risposta corretta

Punteggio ottenuto 1,00 su 1,00



Risposta corretta.

La risposta corretta è:

all of the operations composing it are either completed, or they are undone, as if they had never been executed

Domanda 2 Risposta errata Punteggio ottenuto -0,25 su 1,00	A foreign key in a table
	may not be composed of a single element
	should, in turn, be declared as the primary key of another table
	must belong to the primary key of the table in which it is defined 🗡
	may not be a composite key
	none of the answers are correct

Risposta errata.

La risposta corretta è: should, in turn, be declared as the primary key of another table

Domanda 3 Risposta corretta Punteggio ottenuto 1,00 su 1,00

The SQL command

CREATE TABLE T1
(A1 CHAR(5) NOT NULL,
A2 INTEGER,
A3 CHAR(5) NOT NULL,
PRIMARY KEY (A1, A3),
FOREIGN KEY (A2) REFERENCES T2
ON UPDATE SET NULL);

is incorrect because attribute A2 should be declared NOT NULL

creates a table T1 where, upon each UPDATE operation on T1, the foreign key values are reset to NULL

none of the answers are correct

creates a table T1 whose primary key is (A1, A3, A2)

is incorrect because the attributes composing the primary key should be declared

Risposta corretta.

contiguously

La risposta corretta è: none of the answers are correct

Completo

Punteggio max.:

1,00

The following relations are given (primary keys are underlined):

ATHLETE(ACode, AName, DateOfBirth, CityOfResidence)

TOURNAMENT(TCode, TName, Level, OrganizingAssociation)

RACE(RCode, RName, TCode, Date, CityOfRace)

ATHLETE-PARTICIPATES-IN-RACE(ACode, RCode, Ranking)

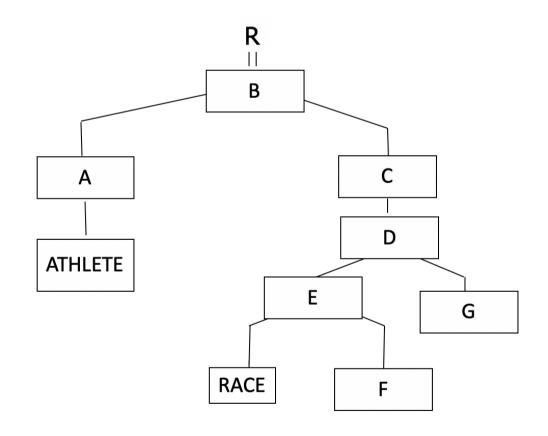
Write the following query in relational algebra:

Show the code and the name of each athlete who has never participated in races held in the city where she/he resides.

Instructions needed to complete the exercise:

The following query tree graphically represents the requested query. You are asked to indicate, for each of the boxes from A to G in the query tree, the relational table or the corresponding algebraic operator. Use the text box below to report the solution.

Note: each box in the query tree is associated with a single relational table or a single algebraic operator.



A, PROJECTION (ACode, AName)

B ANTIJOIN

C SELECTION (CityOfRace=CityOfResidence)

D NATURAL JOIN

E NATURAL JOIN

F ATHLETE-PARTICIPATES-IN-RACES

Completo

1,00

Punteggio max.:

The following relations are given (primary keys are underlined):

DANCER(<u>SSN</u>, Name, DateOfBirth, CityOfResidence, MainDanceType)

DANCE SCHOOL(DSCode, Name, City, ArtisticDirector)

BALLET(<u>BCode</u>, <u>DSCode</u>, Title, #Scene, Type, ChoreographerName)

DANCER-PARTICIPATES-IN-BALLET(<u>SSN</u>, <u>BCode</u>, <u>DSCode</u>, Role)

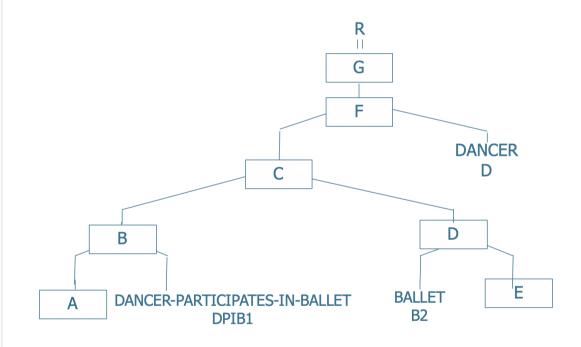
Write the following query in relational algebra:

Show the name and the city of residence of each dancer who has participated in at least two ballets of the same type.

Instructions needed to complete the exercise:

The following query tree graphically represents the requested query. You are asked to indicate, for each of the boxes from A to G in the query tree, the relational table or the corresponding algebraic operator. Use the text box below to report the solution.

Note: each box in the query tree is associated with a single relational table or a single algebraic operator.



A BALLET B1

B NATURAL JOIN

C EQUI JOIN (B1.BCode<>B2.BCode AND B1.Type=B2.Type)

D NATURAL JOIN

E DANCER-PARTICIPATES-IN-BALLET DPIB2

F NATURAL JOIN

G PROJECTION (Name, CityOfResidence)

Completo

Punteggio max.: 1,00

Given the following relational schema

E-BOOK (ISSN, Authorld, Title, Language, Editor)

AUTHOR (Authorld, Name, Surname, Nationality)

DOWNLOAD (Date, Hour, Minutes, Userld, ISSN)

USER (<u>UserId</u>, Name, Gender, DateOfBirth)

For each author of Italian nationality, show name and surname of the author and each editor with which the author has published at least 5 e-books.

SELECT Name, Surname, Editor

FROM AUTHOR A, E-BOOK E

WHERE Nationality="Italian"

AND A.AuthorID= E.AuthorID

GROUP BY Name, Surname, Editor

HAVING COUNT(*)>=5

Domanda 7

Completo

Punteggio max.:

1,00

Given the following relational schema

E-BOOK (ISSN, Authorld, Title, Language, Editor)

AUTHOR (Authorld, Name, Surname, Nationality)

DOWNLOAD (Date, Hour, Minutes, Userld, ISSN)

USER (<u>UserId</u>, Name, Gender, DateOfBirth)

For each editor that has never published e-books written by German authors, show the editor, the total number of published e-books, and the total number of downloads.

```
SELECT EdPub.Editor, Published#, Download#
FROM (SELECT Editor, COUNT(*) AS Published#
FROM E-BOOK
GROUP BY Editor) AS EdPub,
(SELECT Editor, COUNT(*) AS Download#
FROM E-BOOK E2, DOWNLOAD D
WHERE E2.ISSN=D.ISSN
GROUP BY Editor) AS EdDownload
WHERE EdPub.Editor NOT IN
(
SELECT Editor
FROM E-BOOK E1, AUTHOR A1
WHERE Nationality="German" AND E1.AuthorID=A1.AuthorID
)
AND EdPub.Editor=EdDownload.Editor
```

Completo

Punteggio max.: 1,00

Given the following relational schema

E-BOOK (ISSN, Authorld, Title, Language, Editor)

AUTHOR (Authorld, Name, Surname, Nationality)

DOWNLOAD (Date, Hour, Minutes, Userld, ISSN)

USER (<u>UserId</u>, Name, Gender, DateOfBirth)

GROUP BY U.UserID, Name, DateOfBirth

For each user who has downloaded at least 3 different e-books in English, show the name, date of birth and number of (different) editors that published the books he/she downloaded

```
SELECT Name, DateOfBirth, COUNT(DISTINCT Editor) AS Editors#
FROM USER U, DOWNLOAD D, E-BOOK E
WHERE U.UserID IN

(SELECT UserID

FROM(SELECT UserID, COUNT(DISTINCT E1.ISSN) AS
EnglishDifferentBook#

FROM E-BOOK E1, DOWNLOAD D1

WHERE E1.Launguage="English"

AND E1.ISSN=D1.ISSN

GROUP BY UserID)

WHERE EnglishDifferentBook#>=3

)

AND U.UserID=D.UserID AND E.ISSN=D.ISSN
```

Completo

Punteggio max.: 1,00

A gym would like to design a database to manage some of its activities.

Several trainers work at the gym. Each trainer is identified by her/his Social Security Number (SSN) and is characterized by name and surname, date of birth, education, e-mail, phone number (if available) and the list of sports activities the trainer is qualified to teach.

Several sport activities can be carried out at the gym. Activities are identified by a unique code and classified as musical activities, body building, or cardio-fitness. For each musical activity, the name, level (basic, intermediate, advanced) and the list of equipments used (e.g., weights, rope, step) are known. Keep track of the days of the week and of the room where each musical activity takes place, the starting and ending times, and the corresponding trainer. Please note that a trainer can not teach two different musical activities on the same day and time. However, the same musical activity may be carried out at the same time and day in different rooms. Furthermore, the same musical activity can be carried out on different days and many times on the same days in different time periods.

List the main characteristics (Entity Names, their Identifiers, attributes, relationships) of an E-R conceptual schema of the database for the above application.

ENTITY: Trainer

Internal ID: SSN

Attribute: Name, Surname, DateOfBirth, Education, E-mail, Phone#(0,1),

IsQualified(0,1)

ENTITY-FATHER: Activity

Internal ID: CodeA

Attribute: TypeOfActivity

ENTITY-CHILD: Musical-Activity

Attribute: Name, Level

ENTITY: Equipment

Internal ID: Name

ENTITY: EquipForActivity External ID: CodeA, Name

ENTITY: IsQualifiedToTeach

External ID: SSN, CodeA

BINARY-RELATION: DO

Trainer(0,N) - IsQualifiedToTeach(1,1)

IsQualifiedToTeach(SSN) REFERENCES Trainer(SSN)

BINARY-RELATION: ISDONEBY

IsQualifiedToTeach(1,1) - Activity(0,N)

IsQualifiedToTeach(CodeA) REFERENCES Activity(CodeA)

BINARY-RELATION: NEED

EquipForActivity(1,1) - Equipment(0,N)

EquipForAcitivity(1,1)- Activity(0,N)

ENTITY

Domanda 10

Risposta non data

Punteggio max.: 1,00

A gym would like to design a database to manage some of its activities.

Gym members are identified by the number of their membership card. Each member is characterized by a name, surname, date of birth, category (e.g., students, under 16, over 60), and validity time period. Members access the gym. For each access keep track of the corresponding member, date, entrance, and leaving times (e.g., from 10 am to 2 pm). Assume that each member may access the gym only once per date.

The gym offers different packages of activities. Packages are characterized by a code and a brief description. Packages can be purchased by gym members as a subscription, individual entrance, and special mix. For all types of packages, the price and the time slots at which members are allowed to access the gym are known. For subscriptions and special mix, the package duration (in months) is also known. Individual entrances are characterized by the type of entrance (e.g., simple, power).

The database stores all the activity packages purchased by gym members during the year.

Members purchase packages and then associate a time period of usage (starting and ending date) in which the activities of the packages can be carried out at the gym. Assume that a member can associate different packages to the same time period, and the same package can be associated with different members to the same period.

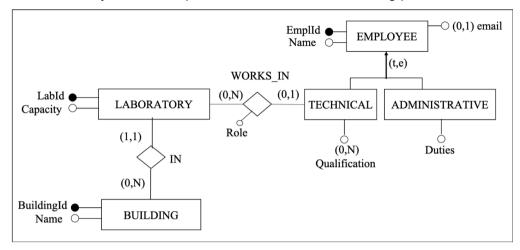
List the main characteristics (Entity Names, their Identifiers, attributes, relationships) of an E-R conceptual schema of the database for the above application.

Completo

Punteggio max.:

1,00

Given the Entity-Relationship schema shown in the following picture



It is asked to:

- 1. Derive a normalized relational logical schema for the same database.
- 2. Define the referential integrity constraints for the 2 relationships defined in the Entity Relationship schema.

BUILDING(BuildingID, Name)

LABORATORY(Labld, Capacity, BuildingID)

EMPLOYEE(EmplId, Name, Email*)

TECNICAL(EmplId)

TECNICAL_ROLE_IN(Emplid, Labid, Role)

ADMINASTRATIVE(EmplId, Duties)

QUALIFICATION(EmplId, NameQualification)

- 1) LABORATORY(BuildingID) REFERENCES BUILDING(BuildingID)
- 2) TECNICAL_ROLE_IN(Emplid) REFERENCES TECNICAL(Emplid)
 TECNICAL_ROLE_IN(Labid) REFERENCES LABORATORY(Labid)

Completo

Punteggio max.:

1,00

The following relations are given (primary keys are underlined).

SUPERMARKET(<u>SupermarketCode</u>, City, NumberOfPlacesInQueue,

OpeningTime, ClosingTime)

ENTRANCE_QUEUE(SupermarketCode, CustomerSSN)

ENTRANCE_REQUEST(<u>RequestCode</u>, SupermarketCode, CustomerSSN, Date, Time)

We would like to automatically manage the entrance queues in a chain of supermarkets.

Write the trigger needed to manage the following activity.

When a customer wants to enter in the queue for a specific supermarket, a new record is inserted into the ENTRANCE REQUEST table.

Write the trigger to check the availability of a place in the queue for the requested supermarket.

The ENTRANCE_QUEUE table contains the customers already queued in each supermarket. Customers are queued until the maximum number of places is reached.

The maximum number of places available in the queue for a supermarket is equal to the value of the NumberOfPlacesInQueue attribute of the SUPERMARKET table. If there is no place available in the queue for the requested supermarket, the trigger ends with an error. Otherwise the customer is put in the queue.

Tip: Use the raise_application_error (....) function to report an error. It is not required to specify the parameters passed to the function.

CREATE OR REPLACE Entrance_Request_trigger
AFTER INSERT ON ENTRANCE_REQUEST
FOR EACH ROW

DECLARE

Max NUMBER;

Counter NUMBER;

BEGIN

SELECT NumberOfPlacesInQueue INTO Max

FROM SUPERMARKET

WHERE SupermarketCode=:NEW.SupermarketCode;

SELECT COUNT(*) INTO Counter

FROM ENTRANCE_QUEUE

WHERE SupermarketCode= :NEW.SupermarketCode;

IF(Max=Counter) THEN	
raise_application_error();	
END IF;	
INSERT INTO ENTRANCE_QUEUE	
VALUES (:NEW.SupermarketCode, :NEW.CustomerSSN);	
END;	

Completo

Punteggio max.: 1,00

The following relations are given (primary keys are underlined).

SUPERMARKET(SupermarketCode, City, NumberOfPlacesInQueue,

OpeningTime, ClosingTime)

ENTRANCE_QUEUE(SupermarketCode, CustomerSSN)

ENTRANCE_REQUEST(<u>RequestCode</u>, SupermarketCode, CustomerSSN, Date, Time)

We would like to automatically manage the entrance queues in a chain of supermarkets.

Write the trigger needed to manage the following activity.

Integrity constraint on the duration of the opening

There can be at most 10 supermarkets in Turin with a duration of the opening (difference between ClosingTime and OpeningTime attributes in the SUPERMARKET table) greater than 18 hours.

Any modification of the SUPERMARKET table that causes the constraint violation must not be executed.

Carefully evaluate all the triggering events on table SUPERMARKET.

Solution guidelines: Given the following template of the trigger solution, you are asked to complete the bold parts (i.e. parts A and B)

create trigger CheckOpeningTime

Α

declare

X number;

begin

В

```
if (X > 10) then
    raise_application_error(...);
end if;
```

end;

A- BEFORE INSERT OR UPDATE ON SUPERMARKET

```
B- SELECT COUNT(*) into X
```

FROM SUPERMARKET

WHERE SupermarketCode<>:NEW.SupermarketCode

AND ClosingTime-OpeningTime>18 AND City="Turin";

```
if(NEW.City="Turin" AND NEW.ClosingTime- NEW.OpeningTime > 18) then
    X= X+1;
```