1. (4 punti) Given the following `CREATE` statements for tables :

```
CREATE TABLE Employee (
cf CHAR(14) PRIMARY KEY,
first_name VARCHAR(50),
last_name VARCHAR(50),
hired_on TIMESTAMP,
email VARCHAR(150) UNIQUE )

CREATE TABLE Project (
number INT NOT NULL,
resp CHAR(14),
budget REAL,
delivery DATE NOT NULL,
PRIMARY KEY (number, resp),
FOREIGN KEY (resp) REFERENCES Manager )
```

```
CREATE TABLE Manager (
cf CHAR(14) PRIMARY KEY,
promoted DATE NOT NULL,
wage REAL,
FOREIGN KEY (cf) REFERENCES Employee )

CREATE TABLE Apprentice (
cf CHAR(14) PRIMARY KEY,
starts DATE NOT NULL,
ends DATE NOT NULL,
wage REAL,
coach CHAR(14),
FOREIGN KEY (cf) REFERENCES Employee,
FOREIGN KEY (coach) REFERENCES Manager )
```

(a) Design the ER schema that produces the above definitions

(b) Can a Manager be also registered as an Apprentice ? ☐ YES ☐ NO ☐ UNDECIDABLE

Non-standard notation (i.e, different from what present in the book and/or in the slides) will be considered a mistake.

DRAW THE ANSWER BELOW:

2. (3 punti) Given the following tables, and with only the tuples present below

`S(A, B, C)`

A	B	C
1	a	a
2	a	a
3	a	c
4	a	e

`T(D, E, F)`

D	E	F
a	e	1
a	c	2
a	a	NULL
b	a	3

Taking into consideration only the tuples showed above, without any further assumption, answer the following **[+1 point for correct answers, −1 points for wrong answers, 0 points if answer left blank]**

(a) Can $S(B, C)$ be foreign key with reference to $T(D, E)$? ☐ YES ☐ NO ☐ UNDECIDIBLE

(b) Can (A, B) be a primary key for S? ☐ YES ☐ NO ☐ UNDECIDIBLE

(c) Can (D, E, F) be a primary key for T? ☐ YES ☐ NO ☐ UNDECIDIBLE

3. (4 punti) Consider the following schema instances (structure and tuples):

`S(A, B)`

A	B
1	10
2	20
3	30

`T(B, C, D)`

B	C	D
1	10	NULL
2	20	f
3	NULL	w

`U(D, E)`

D	E
NULL	f
o	w
f	w
m	NULL

Show the results of the following queries:
WRITE YOUR ANSWER BELOW:

(a)
```
SELECT T.D, SUM(T.B)+1
FROM S JOIN T ON S.B >= T.C+10
GROUP BY T.D
```

(b)
```
SELECT T.D, U.D, U.E
FROM T JOIN U ON T.D=U.E
```

4. (5 punti) Assume you need to design a database for monitoring purchases of customers subscribers of a fidelity card for a grocery chain. The system manages the customers, with their fidelity card account, products that the chain sells, and all the customer's purchases.

Customers are identified by a unique cardID number, then you need to store the first and last name, and the subscription date for the card. The system separates customers with a Premium subscription. For each of them, the system stores the date when the Premium subscription was first registered, and a number of Premium points that the customer acquires in time.

Products are characterized by their name, the VAT number of the producer, the price, and they are identified by their serial code "ProdID".

For each customer, the system keeps track of the list of products bought; each purchase is stored separately. For each purchase, the system also keeps track of date and time of the purchase, amount of products purchased, and price of the single item (*note*: the price may vary over time).

A purchase is identified by the customer, the product and the date and time of the purchase. Only customers with a Premium subscription can buy products with a promotional price. For purchases with a promotional price, the original price is still saved in the system, but addtionally the system stores the discount percentage and the number of premium points acquired by the customer with such purchase.

Provide the following:

(a) the ER diagram for the database to respect all above conditions.

(b) the relational schema of the database (mark also foreign keys in the schema)

Non-standard notation (i.e, different from what present in the book and/or in the slides) will be considered a mistake.

5. (10 punti) A different database stores information about products and reviews, with the following schema:

`Product (code, supplier, name_prod, brand, price, address, discontinued )`
`Review (code, supplier, user, date, name_prod, email, text, rating, brand, price)`

The product is identified by the code and the *id* of the supplier. Of th product the database stores the name, the brand, the price, the address of the supplier and a boolean flag to mark it as discontinued.

The review refers to a single product, contains the id of the user who wrote it, and her email address. It also stores the date in which it has been submitted, the name of the product, the text of the actual review, the rating given to the product, the brand of the product and the price.

Two distinct suppliers can use the same code for any two products even if they are different. A supplier sells only products of a single brand, and has a single address. A product has a single name and price. A user only owns a single email.

(a) Provide in **RELATIONAL ALGEBRA** the query to return code, supplier and brand of products with exactly 2 reviews.

(b) Provide in **RELATIONAL ALGEBRA** the query to return suppliers with all discontinued products.

(c) Provide in **RELATIONAL ALGEBRA** and **SQL** the query to find discontinued products without any review.

(d) Provide in **SQL** the query to find for all suppliers the average number of reviews for products.

(e) Provide in **SQL** the query to find for suppliers with the highest average rating for their products.

6. (2 punti) Given the Relational Schema of Exercise 5:

`Product (code, supplier, name_prod, brand, price, address, discontinued )`
`Review (code, supplier, user, date, name_prod, email, text, rating, brand, price)`
with the same assumptions described above

(a) Find all the functional dependencies

(b) Compute the BCNF decomposition

7. (2 punti) Given the following relations `R(A, B, C, D, E)` e `S(F, G, H)` and the following set of functional dependencies: `AB→C, CD→E, A→E, FG→H` State if any of the following satisfied the above dependencies and as such can be an instance of the schema.

[+1 **point for correct answers,** −1 **points for wrong answers,** 0 **points if answer left blank**]

I.

A	B	C	D	E
1	4	1	3	6
2	4	2	4	7
2	4	2	5	7
2	4	2	6	2

F	G	H
3	2	1
4	5	5

☐ YES ☐ NO

II.

A	B	C	D	E

F	G	H
3	5	1
2	5	1
1	5	1

☐ YES ☐ NO

8. (3 punti) Answer the following questions
[+1 **point for correct answers,** −1 **points for wrong answers,** 0 **points if answer left blank**]

(a) If R and S are relations with only common attribute a, then it holds that $\sigma_{R.a=3}(R \times S) \equiv \sigma_{a=3}(R) \times \sigma_{a=3}(S)$.
☐ YES ☐ NO ☐ UNDECIDABLE.

(b) If the attribute A for the relation R is Foreign Key referencing Y in the relation S, then it holds that A in R can contain `NULL` values .
☐ YES ☐ NO ☐ UNDECIDABLE.

(c) The $\cap$ operator in relational Algebra can be equivalently expressed by using $\bowtie$ and π operators
☐ YES ☐ NO ☐ UNDECIDABLE.

Rispondi alle seguenti domande indicando la risposta corretta / Respond to the following questions indicating the correct answer.

9 To execute the query $\sigma_{A>4}(R)$ it is better to use a table scan on R than to use a hash index on the attribute A of R given the fact that R is clustered. **True or False?**

10 It is possible to have two indexes (on different attributes) on the same relation provided of course that they are both clustered. **True or False?**

11 During query optimization of select-project-join queries the best strategy is to push the selections as early as possible so that we make sure that the we have less tuples to deal with and the overall query will execute faster. **True or False?**

12 Consider relations R[A,B] and S[B,C], and assume that there is at least one tuple in R that has a B-value which is greater than 10. Then, the query "SELECT * FROM R, S WHERE R.B=S.B AND R.B>10" returns at least one tuple. **True or False?**

13 The cardinality of the results of the join of two tables depends on the order the join is performed ($R \bowtie_C S$ or $S \bowtie_C R$). **True or False?**

14 You cannot create two indexes on the same attribute because the database does not know how to arrange the tuples to participate in both indexes. **True or False?**

15 Consider the relation $\sigma_{A=6 \text{ AND } B=4}(R)$. If the reduce factor for A=6 is 0.2 and for B=4 is 0.5, the reduce factor of the selection can be assumed to be 0.1 assuming that there is no functional dependency between A and B and vice versa. **True or False?**

16 A smart optimization of the query SELECT * FROM R WHERE A>5 OR A≤5 is to replace it with the equivalent query SELECT * from R (assuming of course that A is an attribute of R) **True or False?**

Considera lo schema seguente / Consider the following schema

Employee [Eid, EName, Salary, WorksFor]

17 Dept [Did, DName, Budget, Mgr]

Emp.WorksFor è una chiave esportata riferita a Dept.Did / Emp.WorksFor is a foreign key referencing to Dept.Did

Dept.Mgr è una chiave esportata riferita a Emp.Eid / Dept.Mgr is a foreign key referincing to Emp.Eid

Assume different alternatives on what indexes have been created on that schema.

A. clustered B-tree su Emp[Eid] e clustered B-tree su Dept[Did]
B. clustered B-tree su Emp[WorksFor] e clustered B-tree su Dept[Did]
C. clustered hash su Emp[Eid] e clustered hash su Dept[Mgr]
D. clustered B-tree su Emp[Salary] e clustered B-tree su Dept[Did]
E. clustered B-tree su Emp[Salary] e clustered B-tree su Dept[Budget]
F. (nessun indice) / No index at all.

For each query/update, pick the best index design from those above. If two or more designs support the same performance for the given query/update, indicate all these designs. You may use a design multiple times (or not at all). Assume no indexes are present except the ones indicated in each design.

(You have no other information apart from the one presented here)

- Find the Eid's of employees whose salary is greater than the budget of the department with Did = 1
  - **Risposta/Answer:**
- Insert an Employee tuple
  - **Risposta/Answer:**
- Delete all employees in the department with Did = 1
  - **Risposta/Answer:**

Considera il seguente schema:

- Writer[Wid:Integer, Name:VarChar, Age:Integer]
- Paper[Pid:Integer, PubDate:Date, Abstract:VarChar]

18

Assume that every tuple Writer takes 50 bytes, and that every tuple Paper takes 200bytes. There are 4,000 tuples in Writer and 10,000 in Paper. Moreover, a page on disk takes 1000bytes. You can assume that the cost to read a leaf node in the B+Tree takes 4 reads to disk, and that an hash index takes 1,2 (average) accesses to disk.

Consider the following two queries: (q1) "find all writers with age=30" and (q2) "find all papers published on 09/06/2002." Assume the number of qualifying tuples (that is, the number of tuples in the result) is the same for both queries. Assume that you want to create a B+ tree index on age and a B+ tree index on PubDate to speed up these queries. Which of the two choices will be more beneficial to do as a clustered B+tree index as opposed to un-clustered? You can assume that the problems that clustered indexed have are equally bad for both relations. Concentrate only on the benefit that the clustered index can offer. Explain why. Justify your answer.

Risposta/Answer:

Consider the query "find all writers with age>30." Suppose that the Writer relation is clustered and that there is an un-clustered B-tree index on attribute Age. Let the number of qualifying tuples be N. For what values of N is a table-scan of the Writer relation cheaper than using the index? Illustrate apart from the final answer, how you reached that conclusion.

19

Risposta/Answer:

20 Consider relations **R[A,B,C]** and **S[B,D,E]**, and assume that **R** has **K** tuples occupying **M** pages, and **S** has **L** tuples occupying **N** pages. There is a 2-attribute hash index on the attribute pair **(B,D)** of the relation **S**. **B** is the key for **S**. The cost of the best strategy we can do for the join **R and S on the attribute B is:**

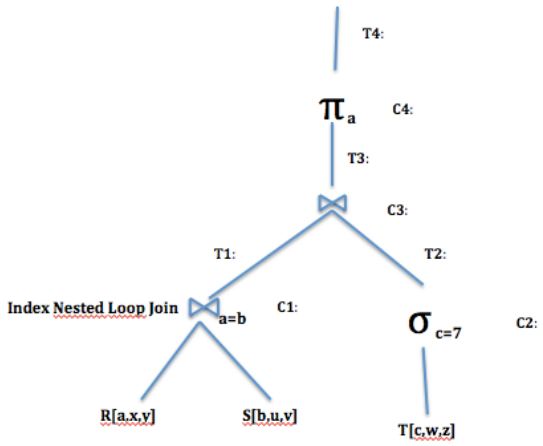
A : M+N
B : M+K*(logN+1)
C : M+M*N
D : M+K*1,2*N
E : M+K*(1,2+1)
F : M+K*2*N*logN

21 **Consider 3 relations R, S and T**

A page is of 750 bytes
R has N_R =10000 tuples each of which is t_R =50 bytes long
S has N_s =500 tuples each of which is t_s =350 bytes long
T has N_T =100K tuples each of which is t_s =1000 bytes long
c is an attribute of T and is an integer
The selectivity factor of the attribute T.c is 25%.
Attribute a is key for R and b is a FK of S referencing R.a
The lookup cost of a B+tree is 4 and of a Hash is 1.2
There are no attributes that can be NULL

Assuming that R and S are ordered on the disk and no index exists, write down the **formula that computes the cost of finding** $R \bowtie_{R.a=S.b} S$ and printing the results on the screen of the user.

22 Assuming that R has a B+tree index on attribute a, and the query execution tree below. For every node of the tree**, compute the Cost of implementing the operation** (noted as C1,C2,C3,C4) and **the tuples produced** (noted as T1,T2,T3,T4). Do not make any calculations. Simply write down the formula. The formula can have only numbers, operations (like + - * /), and variables from the C1, C2, C3,… and from the T1, T2, T3, … etc.



C1 =

T1 =

C2 =

T2 =

C3 =

T3 =

C4 =

T4 =