Started on	Wednesday, 17 May 2023, 09:02
State	Finished
Completed on	Wednesday, 17 May 2023, 12:28
Time taken	3 hours 25 mins

Information

Instructions:

- You have 4 hours (or more if granted to you by SAP) to answer 7 problems described in the following sections.
- Section 1B is only for BSc students and section 1M is only for MSc students.
- With the exception of section 3, all sections have a preamble with the general instructions for that particular section.
- You must provide all your answers online, i.e., directly in this Exam Quiz.

Negative points: in all multiple choice questions, a wrong answer gives negative points so that the sum of all wrong answers cancels out the correct answers. For example, if you have 5 options, two are correct and three are wrong, each wrong answer is worth -33.33%.

IMPORTANT: it is your won responsibility to make sure to SUBMIT and FINISH your attempt: if an exam is not properly submitted we WILL NOT take it into account.

Good luck!

Information

You will work with the database DancingContests (i2dbs-may-2023-DB.sql) found in LearnIT (in the Ordinary Exam Section) using your PostgreSQL installation. The database contains information on awards and ranks in dancing contests with the schema below. Note that the data in the database is entirely fabricated. The database is as follows:

- Contest(id, edition, name, organizer, year)
- Award(id, contestId, description)
- Dancer(id, name, email)
- DancerAward(dancerId, awardId)
- Rank(dancerId, contestId, date, level, rank)

Primary and foreign key attributes are those whose names include Id (but no constraints are defined in the provided database). The first three relations have their first attribute as primary key. The last two relations have a composite primary key consisting of their two first attributes. All relations are self-explanatory. e.g. an award is a recognition given by a contest, e.g., “Most Artistic Dancer”, or “Revelation of the Year’.

Information

To answer this question, you will need to study the schema of the DancingContests database in the "DancingContests Database Description" section.

Consider the following code:

```
DELIMITER $
CREATE PROCEDURE insertDA(theDancerID INTEGER, awardDescription VARCHAR(100))
BEGIN
    DECLARE theAwardID INTEGER;
    SELECT MIN(id) INTO theAwardID
    FROM Award
    WHERE description = awardDescription;
    INSERT INTO DancerAward(dancerID, awardID)
    VALUES (theDancerID, theAwardID);
END
$
DELIMITER ;
CALL insertDA(2, "Revelation of the Year");
```

Question 1

Not answered

Marked out of 5.00

Given the preamble above, select the true statements:

- a. Using the MIN(id) statement ensures that the stored procedure always retrieves the same awardID value.
- b. If dancer with id = 2 does not exist, the stored procedure creates it.
- c. If an award with the given description does not exist, the stored procedure insertDA simply exits without trying to insert an entry into DancerAward.
- d. Using insertDA may not always be the best approach to insert dancer's awards into the database, but it provides encapsulation, reusability, and consistency.
- e. The stored procedure insertDA can be converted to a database function.

Information

To answer this question, you will need to study the schema of the DancingContests database.

Consider the following Java code:

```
public static void deleteDA(Connection conn, int dancerId) throws SQLException {
    try {
      Statement st = conn.createStatement()
      conn.setAutoCommit(false);
      st.execute("DELETE FROM DancerAward WHERE dancerId=" + dancerId);
      st.execute("DELETE FROM Dancers WHERE id=" + dancerId);
      conn.commit();
    } catch (Exception e) {
      conn.rollback();
      throw e;
    } finally {
      st.close();
      conn.setAutoCommit(true);
    }
}
```

Question 2

Complete

Marked out of 5.00

Given the preamble above, select the true statements:

- [ ] a. The code frees the resources associated to the statement when no longer needed.
- [x] b. The connection is closed automatically when a statement is closed.
- [ ] c. The code executes two transactions.
- [x] d. The code is safe against SQL injection attacks.
- [x] e. The code is using transactions correctly.

Information

For each of the following questions on the DancingContests database, select the SQL query and provide the result size (numb records). Note that several SQL queries could be a correct answer for a given question but you must select the one that is the most efficient in general. Also note that queries must return correct results, which might be the empty set (that is, no results at all).

Suggestion: we advise you to come up with your own queries before looking at the different choices to avoid getting confused.

Question 3

Complete

Marked out of 4.00

How many records in DancerAward have an award that does not exist?

a.

```sql
select count(*)
from DancerAward DA
full join Award A on DA.awardId = A.id
where A.id is null;
```

b.

```sql
select count(*)
from DancerAward DA
join Award A on DA.awardId = A.id;
```

c.

```sql
select count(*)
from DancerAward DA
join Award A on DA.awardId = A.id
where A.id is null;
```

d.

```sql
select count(*)
from DancerAward DA
left join Award A on DA.awardId = A.id
where A.id is null;
```

e.

```sql
select count(*)
from DancerAward DA
right join Award A on DA.awardId = A.id
where A.id is null;
```

f.

```sql
select count(*)
from DancerAward DA
right join Award A on DA.awardId = A.id
where A.id = null;
```

Question 4

Complete

Marked out of 1.00

How many results does the correct query return?

Answer: 6

Question 5

Complete

Marked out of 4.00

How many dancers have an email address with hotmail.com?

a.

```sql
select *
from Dancer
where email like '%@hotmail.com';
```

b.

```sql
select *
from Dancer
where email = '@hotmail.com';
```

c.

```sql
select count(*)
from Dancer
where email like '%@hotmail.com';
```

d.

```sql
select *
from Dancer
where email like '@hotmail.com';
```

e.

```sql
select count(*)
from Dancer
where email = '%@hotmail.com';
```

f.

```sql
select count(*)
from Dancer
where email like '@hotmail.com';
```

Question 6

Complete

Marked out of 1.00

How many results does the previous query return?

Answer: 75

Question 7

Complete

Marked out of 4.00

How many distinct dancers have a rank and an award in some contest organized by “DR’?

a.

```sql
select count(distinct Rank.dancerId)
from Rank
join DancerAward DA on Rank.dancerId = DA.dancerId
join Award A on DA.awardId = A.id
where Rank.contestId in (
          select id
          from Contest
          where organizer = 'DR'
);
```

b.

```sql
select count(Rank.dancerId)
from Rank
join DancerAward DA on Rank.dancerId = DA.dancerId
join Award A on DA.awardId = A.id
where Rank.contestId = A.contestId
  and Rank.contestId in (
          select id
          from Contest
          where organizer like 'DR'
  );
```

c.

```sql
select distinct Rank.dancerId
from Rank
join DancerAward DA on Rank.dancerId = DA.dancerId
join Award A on DA.awardId = A.id
where Rank.contestId = A.contestId
  and Rank.contestId in (
          select id
          from Contest
          where organizer like 'DR'
);
```

d.

```sql
select count(Rank.dancerId)
from Rank
join DancerAward DA on Rank.dancerId = DA.dancerId
join Award A on DA.awardId = A.id
where Rank.contestId in (
          select id
          from Contest
          where organizer = 'DR'
);
```

e.

```sql
select count(distinct Rank.dancerId)
from Rank
join DancerAward DA on Rank.dancerId = DA.dancerId
join Award A on DA.awardId = A.id
where Rank.contestId = A.contestId
  and Rank.contestId in (
          select id
          from Contest
          where organizer like 'DR'
  );
```

f.

```sql
select Rank.dancerId
from Rank
join DancerAward DA on Rank.dancerId = DA.dancerId
join Award A on DA.awardId = A.id
where Rank.contestId in (
          select id
          from Contest
          where organizer = 'DR'
);
```

Question 8

Complete

Marked out of 1.00

How many results does the previous query return?

Answer: 223

Question 9

Complete

Marked out of 4.00

How many pairs of contests have the same name?

a.

```sql
select count(*)
from (
    select *
    from Contest C1
    join Contest C2 on C1.id = C2.id
    where C1.name = C2.name
) as Z;
```

b.

```sql
select count(*)
from (
    select count(*)
    from Contest C1
    join Contest C2 on C1.id = C2.id
    where C1.name > C2.name
) as Z;
```

c.

```sql
select count(*)
from (
    select *
    from Contest C1
    join Contest C2 on C1.id > C2.id
    where C1.name = C2.name
) as Z;
```

d.

```sql
select count(*)
from (
    select count(*)
    from Contest C1
    join Contest C2 on C1.id > C2.id
    where C1.name = C2.name
) as Z;
```

e.

```sql
select count(*)
from (
    select *
    from Contest C1
    join Contest C2 on C1.id = C2.id
    where C1.name > C2.name
) as Z;
```

f.

```sql
select count(*)
from (
    select *
    from Contest C1
    join Contest C2 on C1.id > C2.id
    where C1.name > C2.name
) as Z;
```

Question 10

Complete

Marked out of 1.00

How many results does the previous query return?

Answer: 5

Question 11

Complete

Marked out of 4.00

How many distinct dancers have an award, but not a rank, **in a contest**?

a.

```sql
select count(dancerId)
from DancerAward DA
join Award A on DA.awardId = A.id
where not exists (
    select *
    from Rank R
    where DA.dancerId = R.dancerId
    and A.contestId = R.contestId
);
```

b.

```sql
select count(dancerId)
from (
    select DancerAward.dancerId, Rank.contestId as rank,
           Award.contestId as awd
    from DancerAward
    join Award on DancerAward.awardId = Award.Id
    left join Rank on Award.contestId = Rank.contestId
                  and DancerAward.dancerId = Rank.dancerId
) a
where rank is NULL;
```

c.

```sql
select count(distinct dancerId)
from (
    select DancerAward.dancerId, Rank.contestId as rank,
           Award.contestId as awd
    from DancerAward
    left join Award on DancerAward.awardId = Award.Id
    join Rank on Award.contestId = Rank.contestId
              and DancerAward.dancerId = Rank.dancerId
) a
where rank is NULL;
```

d.

```sql
select count(distinct dancerId)
from DancerAward DA
join Award A on DA.awardId = A.id
where not exists (
        select *
    from Rank R
    where DA.dancerId = R.dancerId
            and A.contestId = R.contestId
);
```

e.

```sql
select count(dancerId)
from (
    select DancerAward.dancerId, Rank.contestId as rank,
           Award.contestId as awd
    from DancerAward
    left join Award on DancerAward.awardId = Award.Id
    join Rank on Award.contestId = Rank.contestId
              and DancerAward.dancerId = Rank.dancerId
) a
where rank is NULL;
```

f.

```sql
select count(distinct dancerId)
from (
    select DancerAward.dancerId, Rank.contestId as rank,
           Award.contestId as awd
    from DancerAward
    join Award on DancerAward.awardId = Award.Id
    left join Rank on Award.contestId = Rank.contestId
                   and DancerAward.dancerId = Rank.dancerId
) a
where rank is NULL;
```

Question 12

Complete

Marked out of 1.00

How many results does the previous query return?

Answer: 486

Question 13

Complete

Marked out of 4.00

How many records have a rank lower than the average rank of all records in the relation?

a.

```sql
select count(*)
from Rank
where rank < (select avg(rank) from Rank where level > 1);
```

b.

```sql
select count(level)
from Rank
where rank < (select avg(rank) from Rank where level > 1);
```

c.

```sql
select count(*)
from Rank
where rank < (select avg(rank) from Rank);
```

d.

```sql
select count(level)
from Rank
where rank < (select avg(level) from Rank);
```

e.

```sql
select count(rank)
from Rank
where rank < (select avg(rank) from Rank);
```

f.

```sql
select count(rank)
from Rank
where rank < (select avg(rank) from Rank where level > 1);
```

Question 14

Complete

Marked out of 1.00

How many results does the previous query return?

Answer: 969

Question 15

Complete

Marked out of 4.00

How many dancers have participated in all contests editions named “Dance Forever”?

a.

```sql
select count(*)
from Dancer D
where not exists (
    select *
    from Contest C
    where name = 'Dance Forever'
    and not exists (
        select *
        from Rank R
        where D.Id = R.dancerId
    )
);
```

b.

```sql
select count(dancerId)
from (
    select dancerId
    from Rank
    where contestId in (
        select id
        from Contest
        where name like 'Dance Forever'
    )
    group by dancerId
) X;
```

c.

sql

```sql
select count(dancerId)
from (
    select dancerId
    from Rank
    where contestId in (
        select id
        from Contest
        where name like 'Dance Forever'
    )
    group by dancerId
    having count(contestId) = (
        select count(*)
        from Contest
        where name = 'Dance Forever'
    )
) X;
```

d.

```sql
select count(*)
from (
    select dancerId
    from Rank
    where contestId in (
        select id
        from Contest
        where name like 'Dance Forever'
    )
    group by dancerId
    having count(contestId) = (
        select count(*)
        from Contest
        where name = 'Dance Forever'
    )
) X;
```

e.

```sql
select count(*)
from (
    select dancerId
    from Rank
    where contestId in (
        select id
        from Contest
        where name like 'Dance Forever'
    )
    group by dancerId
) X;
```

f.

```sql
select count(name)
from Dancer D
where not exists (
    select *
    from Contest C
    where name = 'Dance Forever'
    and not exists (
        select *
        from Rank R
        where D.Id = R.dancerId
    )
);
```

Question 16

Complete

Marked out of 1.00

How many results does the previous query return?

Answer: 24

Question 17

Complete

Marked out of 4.00

How many contest names are used by two different organizers?

a.

```sql
select count(*)
from (
    select count(distinct organizer)
    from Contest
    group by name
    having count(distinct organizer) = 2
) X;
```

b.

```sql
select count(*)
from (
    select count(distinct organizer)
    from Contest
    group by organizer
    having count(distinct organizer) = 2
) X;
```

c.

```sql
select count(*)
from (
    select count(distinct organizer)
    from Contest
    group by name
    having count(organizer) = 2
) X;
```

d.

```sql
select count(*)
from (
    select count(distinct organizer)
    from Contest
    group by organizer
    having count(distinct organizer) = 2
) X;
```

e.

```sql
select count(*)
from (
    select *
    from Contest
    group by id
    having count(distinct organizer) = 2
) X;
```

f.

```sql
select count(*)
from (
    select count(distinct organizer)
    from Contest
    group by id
    having count(distinct organizer) = 2
) X;
```

Question 18

Complete

Marked out of 1.00

How many results does the previous query return?

Answer: 5

Question 19

Complete

Marked out of 4.00

Consider a table Person(SSN, ID, Name, Zip, City) with the following dependencies:

- SSN → Name
- SSN,Name → ID
- Zip,City → SSN, ID, Name
- Zip,City → Zip

Normalize Person to BCNF and select the resulting relations:

- [ ] a. Person5(SSN,Name,City)
- [x] b. Person1(Zip,City,SSN,Name)
- [ ] c. Person2(Zip,City,Name)
- [ ] d. Person4(SSN,ID,Zip)
- [x] e. Person3(SSN,ID,Name)

Question 20

Complete

Marked out of 3.00

Consider a table Animal(Especie, Race, Name, Dangerous, Habitat) with the following dependencies:

- Name → Especie
- Race → Dangerous
- Especie → Habitat
- Especie,Race → Name

Select the true statements:

- [ ] a. Especie,Race → Name is an unavoidable functional dependency.
- [ ] b. Especie,Race is the only key of Animal.
- [x] c. The relation Animal2(Especie, Race, Habitat, Dangerous) is not in BCNF.
- [ ] d. Normalizing to 3NF (but not to BCNF) results in exactly two relations.

Question 21

Complete

Marked out of 3.00

Consider a table Car(ID, Make, Model, Size, Type) with the following dependencies:

- ID → Model
- Model → ID
- Size → Type
- ID → Make,Model,Size,Type

Select the true statements:

- [x] a. Normalizing to BCNF results in exactly two relations.
- [ ] b. ID is the only key of Car.
- [x] c. The relation Car2(ID, Make, Model, Size) is in BCNF.
- [ ] d. Make,Model,Size → Size is not a trivial functional dependency.

Information

Consider the following schedules of operations seen by a database management system (time flows from left to right):

Schedule 1

```
Transaction T1:  R(A)  W(B)
Transaction T2:                  R(B)  W(A)
```

Schedule 2

```
Transaction T3:           R(A)              W(B)
Transaction T4:  R(A)            W(B)
```

Schedule 3

```
Transaction T5:  R(A)  R(B)                      W(B)
Transaction T6:                  R(C)  R(B)            W(C)
```

Question 22

Complete

Marked out of 5.00

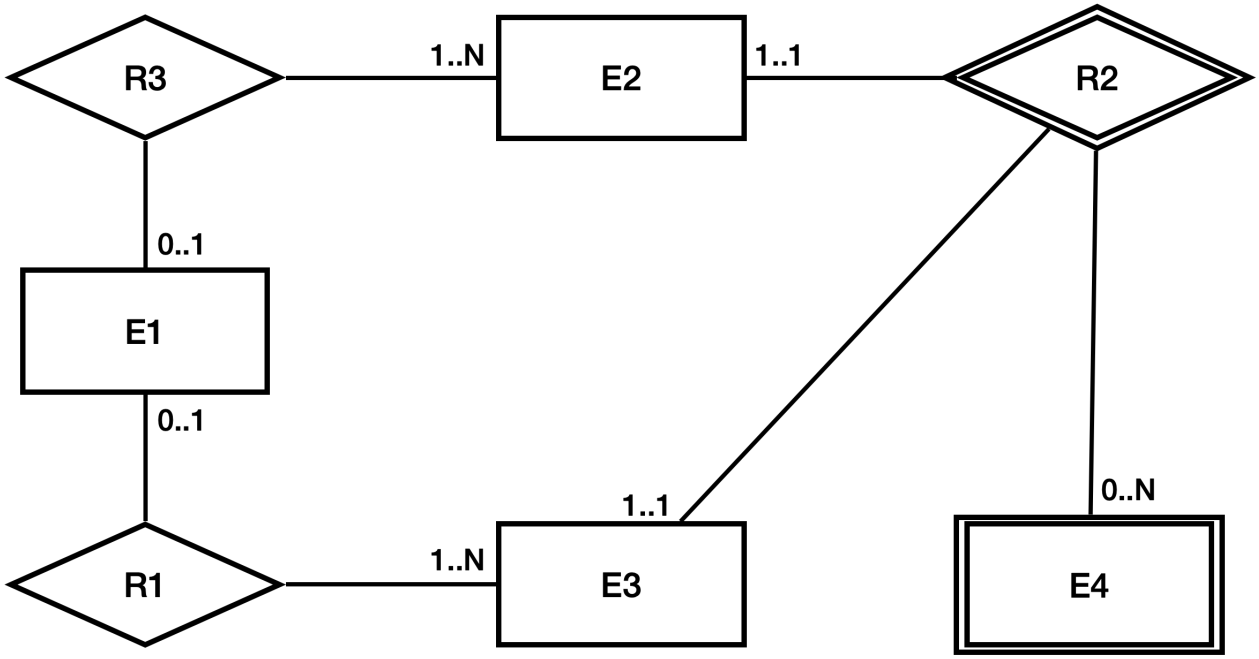
Given the preamble above, select the true statements:

- [ ] a. Schedule 1 is serial.
- [ ] b. Schedule 1 is serializable.
- [ ] c. Schedule 1 is possible with Strict Two-Phase Locking (S2PL).
- [x] d. Schedule 2 is serial.
- [ ] e. Schedule 2 is serializable.
- [ ] f. Schedule 2 is possible with Strict Two-Phase Locking (S2PL).
- [ ] g. Schedule 3 is serial.
- [ ] h. Schedule 3 is serializable.
- [ ] i. Schedule 3 is possible with Strict Two-Phase Locking (S2PL).

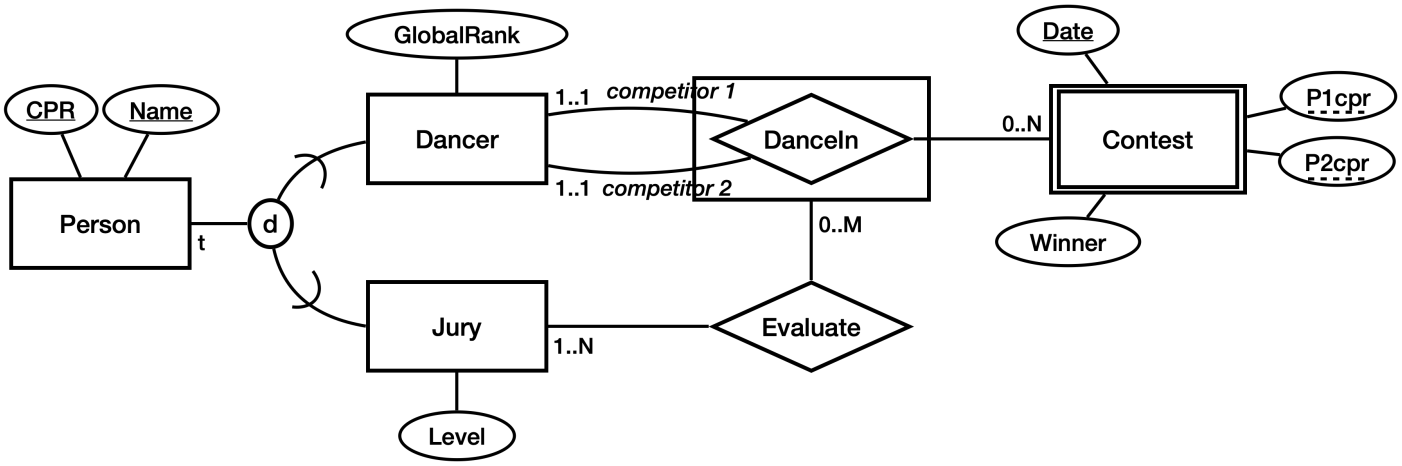
Information

Consider the following ER diagrams:

- ER diagram 1 -- Generic ER Diagram



- ER diagram 2 -- Simplified ER Diagram for a DancerContest database



Suggestion: we advise you to come up with your own DDL queries before looking at the different choices to avoid getting confused.

Question 23

Complete

Marked out of 7.50

Given the ER Diagram 1 in the preamble , select the true statements:

- [ ] a. An E4 is always indirectly related to an E1 through R2 and R3.
- [ ] b. An E2 can be related through R2 to several E3.
- [ ] c. An E4 is uniquely identifiable by its own attributes.
- [x] d. Every E3 entity participates in R2.
- [x] e. Every E1 that participates in R1 is related to at least one E3.

Question 24

Complete

Marked out of 7.50

Given the ER Diagram 2 in the preamble, select the true statements:

- [ ] a. Every contest can have more than two dancers.
- [ ] b. Multiple juries can evaluate the same DanceIn relation.
- [x] c. A jury cannot be a dancer.
- [x] d. A contest is not uniquely identified by its attributes.
- [ ] e. Every dancer dances in at least one contest.

Question 25

Complete

Marked out of 1.00

Select the best DDL query to create the Person relation in the ER Diagram 2. The relations must include all key and foreign key constraints. Make reasonable assumptions on the attributes.

a.

```sql
CREATE TABLE Person (
  CPR INTEGER,
  name VARCHAR(50)
);
```

b.

```sql
CREATE TABLE Person (
  CPR INTEGER,
  name VARCHAR(50) NOT NULL
);
```

c.

```sql
CREATE TABLE Person (
  CPR INTEGER PRIMARY KEY,
  name VARCHAR(50)
);
```

d.

```sql
CREATE TABLE Person (
  CPR INTEGER PRIMARY KEY,
  name VARCHAR(50) NOT NULL
);
```

Question 26

Complete

Marked out of 1.00

Select the best DDL query to create the Dancer relation in the ER Diagram 2. The relations must include all key and foreign key constraints. Make reasonable assumptions on the attributes.

a.

```sql
CREATE TABLE Dancer (
  CPR INTEGER PRIMARY KEY REFERENCES Person (CPR),
  GlobalRank INTEGER NOT NULL
);
```

b.

```sql
CREATE TABLE Dancer (
  CPR INTEGER REFERENCES Person (CPR),
  GlobalRank INTEGER NOT NULL
);
```

c.

```sql
CREATE TABLE Dancer (
  CPR INTEGER PRIMARY KEY REFERENCES Person (CPR),
  GlobalRank INTEGER
);
```

d.

```sql
CREATE TABLE Dancer (
  CPR INTEGER REFERENCES Person (CPR),
  GlobalRank INTEGER
);
```

Question 27

Complete

Marked out of 1.00

Select the best DDL query to create the Jury relation in the ER Diagram 2. The relations must include all key and foreign key constraints. Make reasonable assumptions on the attributes.

a.

```sql
CREATE TABLE Jury (
  CPR INTEGER PRIMARY KEY REFERENCES Dancer (CPR),
  title VARCHAR(50) NOT NULL
);
```

b.

```sql
CREATE TABLE Jury (
  CPR INTEGER PRIMARY KEY REFERENCES Dancer (CPR),
  title VARCHAR(50)
);
```

c.

```sql
CREATE TABLE Jury (
  CPR INTEGER REFERENCES Person (CPR),
  title VARCHAR(50) NOT NULL
);
```

d.

```sql
CREATE TABLE Jury (
  CPR INTEGER PRIMARY KEY REFERENCES Person (CPR),
  title VARCHAR(50) NOT NULL
);
```

Question 28

Complete

Marked out of 1.00

Select the best DDL query to create the DanceIn relation in the ER Diagram 2. The relations must include all key and foreign key constraints. Make reasonable assumptions on the attributes.

a.

```sql
CREATE TABLE DanceIn (
  competitor1 INTEGER REFERENCES Person(CPR),
  competitor2 INTEGER REFERENCES Person(CPR),
  time TIMESTAMP,
  winner BOOLEAN NOT NULL
);
```

b.

```sql
CREATE TABLE DanceIn (
  competitor1 INTEGER REFERENCES Person(CPR),
  competitor2 INTEGER REFERENCES Person(CPR),
  time TIMESTAMP,
  winner BOOLEAN NOT NULL,
  PRIMARY KEY (competitor1, competitor2, time)
);
```

c.

```sql
CREATE TABLE DanceIn (
  competitor1 INTEGER REFERENCES Dancer(CPR),
  competitor2 INTEGER REFERENCES Dancer(CPR),
  time TIMESTAMP,
  winner BOOLEAN NOT NULL,
  PRIMARY KEY (competitor1, competitor2, time)
);
```

d.

sql

```sql
CREATE TABLE DanceIn (
  competitor1 INTEGER REFERENCES Dancer(CPR),
  competitor2 INTEGER REFERENCES Dancer(CPR),
  time TIMESTAMP,
  winner BOOLEAN NOT NULL,
  PRIMARY KEY (competitor1, competitor2)
);
```

Question 29

Complete

Marked out of 1.00

Select the best DDL query to create the Evaluate relation in the ER Diagram 2. The relations must include all key and foreign key constraints. Make reasonable assumptions on the attributes.

a.

```sql
CREATE TABLE Evaluate (
  evaluator integer REFERENCES Person (CPR),
  competitor1 integer,
  competitor2 integer,
  time timestamp,
  primary key (competitor1, competitor2, evaluator, time),
  foreign key (competitor1, competitor2, time)
    references DanceIn_V1 (competitor1, competitor2, time)
);
```

b.

```sql
CREATE TABLE Evaluate (
  evaluator integer REFERENCES Jury (CPR),
  competitor1 integer,
  competitor2 integer,
  time timestamp,
  primary key (competitor1, competitor2, evaluator, time),
  foreign key (competitor1, competitor2, time)
    references DanceIn_V1 (competitor1, competitor2, time)
);
```

c.

sql

```sql
CREATE TABLE Evaluate (
  evaluator integer REFERENCES Jury (CPR),
  competitor1 integer,
  competitor2 integer,
  time timestamp,
  primary key (competitor1, competitor2, evaluator),
  foreign key (competitor1, competitor2, time)
    references DanceIn_V1 (competitor1, competitor2, time)
);
```

d.

sql

```sql
CREATE TABLE Evaluate (
  evaluator integer REFERENCES Person (CPR),
  competitor1 integer,
  competitor2 integer,
  time timestamp,
  primary key (competitor1, competitor2, evaluator),
  foreign key (competitor1, competitor2, time)
    references DanceIn_V1 (competitor1, competitor2, time)
);
```

Information

Consider a MapReduce job with the following mapper and reducer functions:

```
mapper:
   map(relation_file):
      for each line in relation_file:
         record = new Record(line.split)
         if record.year = "2023"
            return(record.company, record.closing_value)

reducer:
   reduce(company, closing_values):
      val = 0
      for each value in closing_values:
         val = val + value
      return(company, val/closing_values.size)
```

Question 30

Complete

Marked out of 5.00

Considering the MapReduce job in the preamble, select the right job's output given the input data below:

company	year	closing_value
google	2022	145
tesla	2023	245
twitter	2023	198
tesla	2020	182
google	2023	320
tesla	2023	228
twitter	2023	202
google	2022	190
google	2023	369
tesla	2023	244

a.
```
+---------+-----+
| google  | 692 |
| google  | 692 |
| tesla   | 717 |
| tesla   | 717 |
| tesla   | 717 |
| twitter | 400 |
| twitter | 400 |
+---------+-----+
```

b.
```
+---------+-----+
|  tesla  | 717 |
|  google | 692 |
| twitter | 400 |
+---------+-----+
```

c.
```
+---------+-----+
|  tesla  | 239 |
|  google | 346 |
| twitter | 200 |
+---------+-----+
```

d.
```
+---------+-----+
| tesla   | 717 |
| tesla   | 717 |
| tesla   | 717 |
| google  | 692 |
| google  | 692 |
| twitter | 400 |
| twitter | 400 |
+---------+-----+
```

e.
```
+---------+-----+
| google  | 692 |
| tesla   | 717 |
| twitter | 400 |
+---------+-----+
```

f.
```
+---------+-----+
| google  | 346 |
| google  | 346 |
| tesla   | 239 |
| tesla   | 239 |
| tesla   | 239 |
| twitter | 200 |
| twitter | 200 |
+---------+-----+
```

g.
```
+---------+-----+
| tesla   | 239 |
| tesla   | 239 |
| tesla   | 239 |
| google  | 346 |
| google  | 346 |
| twitter | 200 |
| twitter | 200 |
+---------+-----+
```

h.

```
+---------+-----+
|  google | 346 |
|  tesla  | 239 |
| twitter | 200 |
+---------+-----+
```

Question 31

Complete

Marked out of 5.00

Select the true statements:

- a. Hadoop MapReduce is more efficient than Apache Spark because Hadoop is mainly a main-memory system.
- b. Apache Wayang is a big data system that decides where to execute a given analytical task by placing each operator on the right data processing platform, such as Spark and Flink.
- c. The 'V' of variety in the Vs of big data stands for the heterogeneity of records within a single dataset.
- d. The BigData wave started with the goal of replacing relational database management systems because they were not scaling for transactional workloads.
- e. The Google File System was designed primarily for managing large datasets in a distributed environment, including handling failures and providing high throughput.
- f. Apache Spark implements an eager execution model where an analytical job starts to get processed as soon as it is submitted regardless of the operators it contains.
- g. MapReduce is a programming paradigm inspired from functional programming where one implements an analytical task using mainly a Map and a Reduce function.
- h. Cross-platform data processing allows applications to run queries/tasks over multiple data sources and processing platforms seamlessly.

Information

Consider the following relation with information on juries of dancing contests:

- Jury(id, name, age, weight, ...)

Select the indexes (could be 0, 1, or more) that a query optimiser would use to process each of the queries below. Assume a data distribution and selectivity such that whenever a relevant index is available an index-scan is better than a full scan. Indicate whether one of the selected indexes is covering (by selecting the "It is covering" check box).

Question 32

Complete

Marked out of 2.50

Select the relevant index(es) for the query below:

```
select id, name
from Jury
where age = 42;
```

- [ ] a. Jury(id)
- [x] b. Jury(age)
- [ ] c. Jury(weight)
- [ ] d. Jury(age, weight)
- [ ] e. Jury(age, weight, name)
- [ ] f. It is covering
- [ ] g. No index

Question 33

Complete

Marked out of 2.50

Select the relevant index(es) for the query below:

```
select name
from Jury
where age > (select min(weight) from Jury);
```

- [ ] a. Jury(id)
- [ ] b. Jury(age)
- [ ] c. Jury(weight)
- [ ] d. Jury(age, weight)
- [x] e. Jury(age, weight, name)
- [x] f. It is covering
- [ ] g. No index

Question 34

Complete

Marked out of 3.00

Select the relevant index(es) for the query below:

```
select weight
from Jury
where age > 42;
```

- [ ] a. Jury(id)
- [x] b. Jury(age)
- [x] c. Jury(weight)
- [ ] d. Jury(age, weight)
- [ ] e. Jury(age, weight, name)
- [x] f. It is covering
- [ ] g. No index

Question 35

Complete

Marked out of 2.00

Select the relevant index(es) for the query below:

```
select *
from Jury;
```

- [ ] a. Jury(id)
- [ ] b. Jury(age)
- [ ] c. Jury(weight)
- [ ] d. Jury(age, weight)
- [ ] e. Jury(age, weight, name)
- [ ] f. It is covering
- [x] g. No index