

Instructions:

- The usage of books, notes, old exams, and other offline/online resources is explicitly **FORBIDDEN** during the exam. The use of electronic aids such as smart-phones, etcetera, is **ALSO NOT** allowed. Failure to comply with these rules leads to a **FAIL** of the resit.
- This exam contains 20 multiple-choice questions (plus the honour code question at the begin of the exam), randomly selected from several question pools addressing all the topics covered in the course.
- The 20 questions are offered to you in random order, but with a pre-defined order of topics:
 - Knowledge questions on the Relational Model (2 minutes)
 - Knowledge questions on SQL (2 minutes)
 - Knowledge questions on NoSQL (3minutes)
 - Knowledge questions on Neo4J (2 minutes)
 - Exercise questions on the Relational Model (4 minutes)
 - Exercise questions on SQL (20 minutes)
 - Exercise questions on Neo4J (20 minutes)
- The quiz interface of Brightspace will allow **only to move forward** between questions, never backward.
- The exam duration is exactly 60 Minutes, unless you have permission for extra time. This means that your answer sheets **must be handed in not later than 60 Minutes from the official starting time**.
- On average, you have 3 minutes available for each question. In practice, some questions will be very quick to answer (up to 1 min.), others will require more time (up to 5 min.).
- Multiple choice questions are worth 0.5 points each. The total number of possible points is 10.
- There is only one right answer for each multiple-choice question. If you think there is more, pick **the best one**.
- In questions related to SQL (including query executions): the reference version of SQL is **SQL-1999**, and the reference relational database system is **PostgreSQL** version 12 (the one used for the lab assignments).
- In questions related to **Neo4J**: the reference version of **Neo4J** is **v3 Community Edition**. (the one used for the lab assignments).
- In questions related to **MongoDB**: the reference version of **MongoDB** is **v4** (the one referenced in the lecture notes).

Good Luck!

[Page left intentionally blank]

Question 1

Which of the following SQL set-operators cannot be expressed using other constructs of the SQL language?

- A. INTERSECT
- B. UNION**
- C. EXCEPT
- D. MINUS

Question 2

Which of the following are elements of an SQL schema?

- [1] Catalog
 - [2] Custom Constrain Types
- A. None**
 - B. Only [1]
 - C. Only [2]
 - D. Both [1] and [2]

Question 3

Which of the following are elements of an SQL schema?

- [1] Views
 - [2] Indexes
- A. None of the statements is correct
 - B. Only [1]
 - C. Only [2]
 - D. Both statements are correct**

Question 4

Which of the following statements about relations in the relational data model is **correct**?

- [1] Tuples in a relation are, by default, ordered according to their order of insertion
 - [2] An n-tuple is an ordered list of n values
- A. None of the statements is correct
 - B. Only [1]
 - C. Only [2]**
 - D. Both statements are correct

Question 5

Which of the following statements about relations in the relational data model are **correct**?

- [1] NULL represent an attribute value that might be unknown
 - [2] Each value in an n-tuple is *atomic*, that is, it is divisible into sub-atomic components
- A. None of the statements is correct
 - B. Only [1]**
 - C. Only [2]
 - D. Both statements are correct

Question 6

Which of the following statements about relations in the relational data model are **correct**?

- [1] Each value in an n-tuple is *atomic*, that is, it is divisible into sub-atomic components
- [2] Tuples in a relation are, by default, ordered according to their order of insertion

A. None of the statements is correct

B. Only [1]

C. Only [2]

D. Both statements are correct

Question 7

Which of the following statements about *constraints* in the relational data model are **correct**?

- [1] Constraints contribute to data quality and correctness
- [2] Constraints can play a role in query processing optimisation

A. None of the statements is correct

B. Only [1]

C. Only [2]

D. Both statements are correct

Question 8

Which of the following statements about *constraints* in the relational data model are **correct**?

- [1] *Implicit* constraints like *Tuple Constraints* are expressed in the relational schema
- [2] *Implicit* constraints like *Domain Constraints* are implemented in all commercial RDBMs

A. None of the statements is correct

B. Only [1]

C. Only [2]

D. Both statements are correct

Question 9

Which of the following statements about commercial relational databases (e.g. Oracle, MySQL, SQLServer) is **not correct**?

A. Commercial RDBMs are complex systems with features that are not needed in all use cases

B. Horizontal scalability can be easily achieved with commercial RDBM systems

C. Commercial RDBMs manage concurrency in an efficient and effective way

D. Commercial RDBMs typically feature ACID properties

Question 10

Which of the following statements about commercial relational databases (e.g. Oracle, MySQL, SQLServer) is **correct**?

A. Commercial RDBMs are complex systems with rich functionalities that are always compatible across vendors

B. Commercial RDBMS manage concurrency in an efficient and effective way

- C. Horizontal scalability can be easily achieved with commercial RDBMS
- D. Commercial RDBMS typically feature BASE properties

Question 11

Which of the following statements about NoSQL systems is **not correct**?

- A. Key-value stores favour high scalability over consistency
- B. Column Store databases belongs to the same family of databases as Relational Column Databases**
- C. Graph Store databases are designed to handle relationships in an effective manner
- D. Document store databases are similar to Key-value stores, but they allow complex data structures in the *value*

Question 12

Which of the following statements about NoSQL systems is **correct**?

- A. Column Store databases belongs to the same family of databases as Relational Column Databases
- B. Graph Store databases are designed to handle relationships in an effective manner**
- C. Key-value stores typically provide querying and analytics features similar to Relational Column Databases
- D. Document Store databases are designed for transactionality and isolation

Question 13

Which of the following statements about MongoDB systems is **not correct**?

- A. MongoDB features a mechanism for referencing (linking) documents
- B. In MongoDB, equality of non-literal values cannot be tested with standard query operators**
- C. MongoDB features query planning mechanisms
- D. In MongoDB, queries support projection and sorting operations as in SQL

Question 14

Which of the following statements about MongoDB Aggregation framework is **not correct**?

- A. MongoDB Aggregation framework is needed to “join” documents across collections
- B. The match operator is equivalent to the SELECT clause in SQL**
- C. The lookup operator is similar to a LEFT OUTER JOIN clause in SQL
- D. The group operator is similar to a GROUP BY clause in SQL

Question 15

Which of the following statements about views in SQL is **not correct**?

- A. Views can be useful to create more readable queries
- B. Views can be useful to limit users’ visibility on the database schema
- C. View materialisation mechanisms can speed up query execution time
- D. Views can always be used for modifying and deleting database data**

Question 16

Which of the following statements about Triggers in SQL is **not correct**?

- A. Triggers are a mechanism for semantic constraint enforcement
- B. Triggers are a robust integrity checking mechanism to foreign keys
- C. Well-specified Triggers must have an *Event* and an *Action* component
- D. The *Condition* component of Triggers is mandatory for statement-level Triggers**

Question 17

Which of the following statements about Neo4J is **not correct**?

- A. Views can be useful to create more readable queries
- B. Neo4J does not allow the definition of VIEWS
- C. Neo4J allows the specification of Node Keys
- D. Neo4J allows the specification of unique property constraints on relationships**

Question 18

Which of the following statements about Neo4J is **correct**?

- A. Neo4J has specific commands for Triggers
- B. Neo4J has specific commands for the definition of VIEWS
- C. Neo4J allows relationships with multiple types
- D. Neo4J does not allow the specification of unique property constraints on relationships**

Question 19

Which of the following is **not a valid** CYPHER clause?

- [1] HAVING
- [2] UNION
- A. None
- B. Only [1]**
- C. Only [2]
- D. Both [1] and [2]

Question 20

Which of the following statements about the `OPTIONAL MATCH` CYPHER clause is **correct**?

- [1] The `OPTIONAL MATCH` could be considered equivalent to the `INNER JOIN` clause in SQL
- [2] The `OPTIONAL MATCH` can be used to match both nodes and relationships
- A. None
- B. Only [1]
- C. Only [2]**
- D. Both [1] and [2]

Question 21

Consider the following relational schema. Attributes with the same name represent keys and foreign keys (e.g. WorkshopName, AutoLicense).

AutoWorkshop (WorkshopName, Address, Director)

Repair (WorkshopName, ReceiptNumber, AutoLicense, Type, Date, Cost)

Auto (AutoLicense, Owner)

Which of the following statement(s) about the above relational schema is/are **correct**?

- [1] The same Auto can be repaired multiple times by the same AutoWorkshop in the same day
- [2] Two different AutoWorkshops can perform a repair having the same ReceiptNumber
- [3] Two Repairs of the same Type performed on the same Auto by the same AutoWorkshop must have the same cost

A. Only [1] and [2]

B. Only [1] and [3]

C. Only [2] and [3]

D. All the statements are correct

Solution:

[[1]] is correct

[[2]] is correct

[[3]] is a business constraint, it cannot be enforced by the database.

Question 22

Consider the following relational schema. Attributes with the same name represent keys and foreign keys (e.g. WorkshopName, AutoLicense).

AutoWorkshop (WorkshopName, Address, Director)

Repair (WorkshopName, ReceiptNumber, AutoLicense, Type, Date, Cost)

Auto (AutoLicense, Owner)

Which of the following statement(s) about the above relational schema is/are **not correct**?

- [1] The same Auto can be repaired multiple times by the same AutoWorkshop
- [2] Two different Repairs performed by the same AutoWorkshop cannot have the same ReceiptNumber
- [3] Two Repairs of the same Type performed on the same Auto by the same AutoWorkshop must have the same cost

A. Only [1] and [2]

B. Only [1] and [3]

C. Only [2] and [3]

D. All the statements are correct

Solution:

[[1]] is correct

[[2]] is not correct, the primary key of **Repair** include also Workshop Name and Auto-License

[[3]] is a business constraint, it cannot be enforced by the database.

Question 23

Which of the following statements about the miniworld modelled by the **Film** database schema in Appendix A are **not correct**?

- [1] An **actor** can play multiple roles in the same **movie**
 - [2] An instance of the **category** relation must be related to at least one instance from the **film_category** relation
 - [3] A **Comedy** film cannot have an **NC-17 rating**
- A. Only [1] and [2]
B. Only [1] and [3]
C. Only [2] and [3]
D. All the statements are not correct

Solution:

[[1]] is wrong because the **film_category** table has both the **film_id** and the **category_id** as part of the primary key.

[[2]] the foreign constraint works the other way around.

[[3]] is a business constraint, it cannot be enforced by the database.

Question 24

Which of the following statements about the miniworld modelled by the **Film** database schema in Appendix A are **not correct**?

- [1] An **actor** cannot play multiple roles in the same **movie**
 - [2] An instance of the **film** relation must be related to at least one instance from the **film_category** relation
 - [3] A **Classic** film cannot have an **NC-17 rating**
- A. Only [1] and [2]
B. Only [1] and [3]
C. Only [2] and [3]
D. All the statements are not correct

Solution:

[[1]] is correct because the **film_category** table has both the **film_id** and the **category_id** as part of the primary key.

[[2]] is wrong, because the foreign key constraints work in the opposite direction

[[3]] is a business constraint, it cannot be enforced by the database.

Question 25

Consider the database state in Appendix B, and the following three queries:

```

1 SELECT actor_id
2 FROM actor
3 WHERE actor_id NOT IN (SELECT actor_id FROM film_actor)

```

Listing 1: SQL-1

```

1 SELECT *
2 FROM actor, film_actor
3 WHERE actor.actor_id <> film_actor.actor_id

```

Listing 2: SQL-2

```

1 SELECT actor_id
2 FROM actor WHERE actor_id NOT IN (
3 SELECT actor_id FROM film_actor
4 GROUP BY actor_id HAVING count(*) > 0)

```

Listing 3: SQL-3

Which of the following answers correctly describes the output of the three queries?

- A. Only the query in Listing 1 and the query in Listing 2 return the same number of results
- B. Only the query in Listing 1 and the query in Listing 3 return the same number of results**
- C. Only the query in Listing 2 and the query in Listing 3 return the same number of results
- D. All queries return the same number of results

Solution: Query Listing 1

```

1 SELECT actor_id
2 FROM actor
3 WHERE actor_id NOT IN (SELECT actor_id FROM film_actor)

```

actor_id	[PK] integer
1	9

Query Listing 2

```

1 SELECT *
2 FROM actor, film_actor
3 WHERE actor.actor_id <> film_actor.actor_id

```

actor_id	first_name	last_name	actor_id	film_id
1	Ed	Chase	1	1
2	Jennifer	Davis	1	1
3	Johnny	Lithbridge	1	1
4	Bette	Nicholson	1	1
5	Grace	Miscell	1	1
6	Matthew	Johansson	1	1
7	Joe	Swank	1	1
8	Christian	Cable	1	1
9	Ed	Chase	1	3
10	Jennifer	Davis	1	3
11	Johnny	Lithbridge	1	3
12	Bette	Nicholson	1	3
13	Grace	Miscell	1	3

Query Listing 3

```

1 SELECT actor_id
2 FROM actor WHERE actor_id NOT IN (
3 SELECT actor_id FROM film_actor
4 GROUP BY actor_id HAVING count(*) > 0)

```

actor_id	[PK] integer
1	9

Question 26

Consider the database state in Appendix B, and the following three queries:

```

1 SELECT actor_id
2 FROM actor
3 WHERE actor_id NOT IN (SELECT actor_id FROM film_actor)

```

Listing 4: SQL-4

```

1 SELECT *
2 FROM actor, film_actor
3 WHERE actor.actor_id <> film_actor.actor_id

```

Listing 5: SQL-5

```

1 SELECT actor_id
2 FROM actor WHERE actor_id NOT IN (
3 SELECT actor_id FROM film_actor
4 GROUP BY actor_id HAVING count(*) = 0)

```

Listing 6: SQL-6

Which of the following answer correctly describes the output of the three queries?

- A. Only the query in Listing 4 and the query in Listing 5 return the same number of results
- B. Only the query in Listing 4 and the query in Listing 6 return the same number of results
- C. Only the query in Listing 5 and the query in Listing 6 return the same number of results
- D. All queries return a different number of results**

Solution: Query Listing 4

```

1 SELECT actor_id
2 FROM actor
3 WHERE actor_id NOT IN (SELECT actor_id FROM film_actor)

```

actor_id	[PK] integer
1	9

Query Listing 5

```

1 SELECT *
2 FROM actor, film_actor
3 WHERE actor.actor_id <> film_actor.actor_id

```

actor_id	first_name	last_name	actor_id	film_id
integer	character varying (45)	character varying (45)	smallint	smallint
1	Ed	Chase	1	1
2	Jennifer	Davis	1	1
3	Johnny	Lublingda	1	1
4	Bette	Nicholson	1	1
5	Grace	Musiel	1	1
6	Matthew	Johannson	1	1
7	Joe	Stewak	1	1
8	Christian	Gable	1	1
9	Ed	Chase	1	3
10	Jennifer	Davis	1	3
11	Johnny	Lublingda	1	3
12	Bette	Nicholson	1	3
13	Grace	Musiel	1	3

Query Listing 6

```

1 SELECT actor_id
2 FROM actor WHERE actor_id IN (
3 SELECT actor_id FROM film_actor
4 GROUP BY actor_id HAVING count(*) = 0)

```

actor_id	[PK] integer
----------	--------------

Question 27

Consider the database state in Appendix B, and the following two queries:

```
1 SELECT actor_id FROM actor
2 UNION
3 SELECT category_id FROM category
4 INTERSECT
5 SELECT film_id FROM film
```

Listing 7: SQL-7

```
1 SELECT actor_id FROM actor
2 INTERSECT
3 SELECT category_id FROM category
4 UNION
5 SELECT film_id FROM film
```

Listing 8: SQL-8

Which of the following answer correctly describes the output of the two queries?

- A. The two queries return the same number of results.
- B. The query in Listing 7 returns more results than the query in Listing 8**
- C. The query in Listing 8 returns more results than the query in Listing 7
- D. Both queries are syntactically incorrect.

Solution: Query Listing 7

Query Editor		Query History	
1	SELECT	actor_id	FROM actor
2	UNION		
3	SELECT	category_id	FROM category
4	INTERSECT		
5	SELECT	film_id	FROM film
6			
7			

Data Output		Explain	Messages	Notifications
	actor_id integer			
1	5			
2	4			
3	6			
4	2			
5	7			
6	1			
7	8			
8	9			
9	3			

Query Listing 8

Query Editor		Query History	
1	SELECT	actor_id	FROM actor
2	INTERSECT		
3	SELECT	category_id	FROM category
4	UNION		
5	SELECT	film_id	FROM film
6			
7			

Data Output		Explain	Messages	Notifications
	actor_id integer			
1	5			
2	4			
3	6			
4	2			
5	7			
6	1			
7	8			
8	3			

Question 28

Consider the database state in Appendix B, and the following two queries:

```
1 SELECT actor_id FROM actor
2 UNION ALL
3 SELECT film_id FROM film
4 INTERSECT
5 SELECT category_id FROM category
```

Listing 9: SQL-9

```
1 SELECT actor_id FROM actor
2 UNION ALL
3 SELECT category_id FROM category
4 INTERSECT
5 SELECT film_id FROM film
```

Listing 10: SQL-10

Which of the following answer correctly describes the output of the two queries?

- A. The two queries return the same number of results.**
- B. The query in Listing 9 returns more results than the query in Listing 10
- C. The query in Listing 10 returns more results than the query in Listing 9
- D. Both queries are syntactically incorrect.

The screenshot shows a database query interface. At the top, the SQL query for Listing 9 is displayed: `SELECT actor_id FROM actor UNION ALL SELECT film_id FROM film INTERSECT SELECT category_id FROM category`. Below the query, there are tabs for 'Data Output', 'Explain', 'Messages', and 'Notifications'. The 'Data Output' tab is selected, showing a table with the column 'actor_id' (integer) and 14 rows of data. The data values are: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 3, 5, 4, 2, 1.

actor_id integer
1
2
3
4
5
6
7
8
9
10
3
5
4
2
1

Solution: Query Listing 9

Query Editor		Query History
1	SELECT actor_id FROM actor	
2	UNION ALL	
3	SELECT category_id FROM category	
4	INTERSECT	
5	SELECT film_id FROM film	
Data Output		Explain Messages Notifications
actor_id	integer	
1	1	
2	2	
3	3	
4	4	
5	5	
6	6	
7	7	
8	8	
9	9	
10	3	
11	5	
12	4	
13	2	
14	1	

Query Listing 10

Question 29

Consider the database state in Appendix B, and the following query

```

1 SELECT FA1.film_id, count(*)
2 FROM film AS F1 JOIN film_actor AS FA1 ON F1.film_id = FA1.film_id
3     JOIN film_actor AS FA2 ON FA2.film_id = FA1.film_id
4 WHERE FA1.actor_id > FA2.actor_id
5 GROUP BY FA1.film_id
6 ORDER BY FA1.film_id ASC

```

Listing 11: SQL-11

Which of the following answer correctly describes the output of the query?

- A. {< 1, 6 >, < 2, 3 >, < 3, 6 >, < 4, 1 >, < 5, 3 >, < 6, 3 >, < 7, 1 >, < 8, 1 >}
- B. {< 1, 12 >, < 2, 6 >, < 3, 12 >, < 4, 2 >, < 5, 6 >, < 6, 6 >, < 7, 2 >, < 8, 2 >}
- C. {< 1, 4 >, < 2, 3 >, < 3, 4 >, < 4, 2 >, < 5, 3 >, < 6, 3 >, < 7, 2 >, < 8, 2 >}
- D. {NULL}

Query Editor		Query History
1	SELECT FA1.film_id, count(*)	
2	FROM film AS F1 JOIN film_actor AS FA1 ON F1.film_id = FA1.film_id	
3	JOIN film_actor AS FA2 ON FA2.film_id = FA1.film_id	
4	WHERE FA1.actor_id > FA2.actor_id	
5	GROUP BY FA1.film_id	
6	ORDER BY FA1.film_id ASC	
Data Output		Explain Messages Notifications
film_id	count	
1	6	
2	3	
3	6	
4	1	
5	3	
6	3	
7	1	
8	1	

Solution:

Question 30

Consider the database state in Appendix B, and the following query

```

1 SELECT FA1.film_id, count(*)
2 FROM film AS F1 JOIN film_actor AS FA1 ON F1.film_id = FA1.film_id
3      JOIN film_actor AS FA2 ON FA2.film_id = FA1.film_id
4 WHERE FA1.actor_id <> FA2.actor_id
5 GROUP BY FA1.film_id
6 ORDER BY FA1.film_id ASC

```

Listing 12: SQL-12

Which of the following answer correctly describes the output of the query?

- A. {< 1, 6 >, < 2, 3 >, < 3, 6 >, < 4, 1 >, < 5, 3 >, < 6, 3 >, < 7, 1 >, < 8, 1 >}
- B. {< 1, 12 >, < 2, 6 >, < 3, 12 >, < 4, 2 >, < 5, 6 >, < 6, 6 >, < 7, 2 >, < 8, 2 >}**
- C. {< 1, 4 >, < 2, 3 >, < 3, 4 >, < 4, 2 >, < 5, 3 >, < 6, 3 >, < 7, 2 >, < 8, 2 >}
- D. {NULL}

Query Editor Query History

```

1 SELECT FA1.film_id, count(*)
2 FROM film AS F1 JOIN film_actor AS FA1 ON F1.film_id = FA1.film_id
3      JOIN film_actor AS FA2 ON FA2.film_id = FA1.film_id
4 WHERE FA1.actor_id <> FA2.actor_id
5 GROUP BY FA1.film_id
6 ORDER BY FA1.film_id ASC

```

Data Output Explain Messages Notifications

	film_id smallint	count bigint
1	1	12
2	2	6
3	3	12
4	4	2
5	5	6
6	6	6
7	7	2
8	8	2

Solution:

Question 31

Consider the database state in Appendix B, and the following query

```

1 SELECT FA1.film_id, count(*)
2 FROM film AS F1 JOIN film_actor AS FA1 ON F1.film_id = FA1.film_id
3      JOIN film_actor AS FA2 ON FA2.film_id = FA1.film_id
4 WHERE FA1.actor_id = FA2.actor_id
5 GROUP BY FA1.film_id
6 ORDER BY FA1.film_id ASC

```

Listing 13: SQL-13

Which of the following answer correctly describes the output of the query?

- A. {< 1, 6 >, < 2, 3 >, < 3, 6 >, < 4, 1 >, < 5, 3 >, < 6, 3 >, < 7, 1 >, < 8, 1 >}
- B. {< 1, 12 >, < 2, 6 >, < 3, 12 >, < 4, 2 >, < 5, 6 >, < 6, 6 >, < 7, 2 >, < 8, 2 >}
- C. {< 1, 4 >, < 2, 3 >, < 3, 4 >, < 4, 2 >, < 5, 3 >, < 6, 3 >, < 7, 2 >, < 8, 2 >}**
- D. {NULL}

Query Editor Query History

```

1 SELECT FA1.film_id, count(*)
2 FROM film AS F1 JOIN film_actor AS FA1 ON F1.film_id = FA1.film_id
3     JOIN film_actor AS FA2 ON FA2.film_id = FA1.film_id
4 WHERE FA1.actor_id = FA2.actor_id
5 GROUP BY FA1.film_id
6 ORDER BY FA1.film_id ASC

```

Data Output Explain Messages Notifications

	film_id smallint	count bigint
1	1	4
2	2	3
3	3	4
4	4	2
5	5	3
6	6	3
7	7	2
8	8	2

Solution:

Question 32

Consider the database state in Appendix B, and the following query

```

1 SELECT *
2 FROM film JOIN actor
3 WHERE title LIKE 'D%a%' AND actor.last_name LIKE 'G%'

```

Listing 14: SQL-14

Which of the following answer correctly describes the output of the query?

- A. The database returns an error**
- B. The database returns 4 tuples
- C. The database returns 2 tuples
- D. The database returns 0 tuples

Query Editor Query History

```

1 SELECT *
2 FROM film JOIN actor
3 WHERE title LIKE 'D%a%' AND actor.last_name LIKE 'G%'

```

Data Output Explain Messages Notifications

ERROR: syntax error at or near "WHERE"
 LINE 3: WHERE title LIKE 'D%a%' AND actor.last_name LIKE 'G%'
 ^
 SQL state: 42601
 Character: 32

Solution:

Question 33

Consider the database state in Appendix B, and the following query

```

1 SELECT *
2 FROM film, actor
3 WHERE title LIKE 'D%' OR LIKE '%B%'

```

Listing 15: SQL-15

Which of the following answer correctly describes the output of the query?

A. The database returns an error

- B. The database returns 36 tuples
- C. The database returns 72 tuples
- D. The database returns 0 tuples

Solution:

```

1  SELECT *
2  FROM film, actor
3  WHERE title LIKE 'D%' OR LIKE 'B%'

```

Data Output Explain **Messages** Notifications

ERROR: type "like" does not exist
 LINE 3: WHERE title LIKE 'D%' OR LIKE 'B%'
 ^
 SQL state: 42704
 Character: 53

0.1 Schema Change**Question 34**

The database administrator submits the following statement to the database:

```

1  ALTER DOMAIN year ADD CONSTRAINT
2  new_year_check CHECK (VALUE >= 2010 AND VALUE <= 2012);

```

Listing 16: SQL-16

What is the response of the RDBMS?

- A. The database returns an error, because the table film contains values that violate the new constraint**
- B. The database returns an error, because the syntax of the statement is not correct
- C. The database accepts the schema change, and no changes happen to the database state
- D. The database accepts the additional constraint, but all the tuples in the film table are deleted to avoid integrity issues

Solution:

```

1  ALTER DOMAIN year ADD CONSTRAINT new_year_check CHECK (VALUE >= 2010 AND VALUE <= 2012);

```

Data Output Explain **Messages** Notifications

ERROR: column "release_year" of table "film" contains values that violate the new constraint
 SQL state: 23514

Question 35

The database administrator submits the following statement to the database:

```

1  DROP TYPE mpaa_rating;

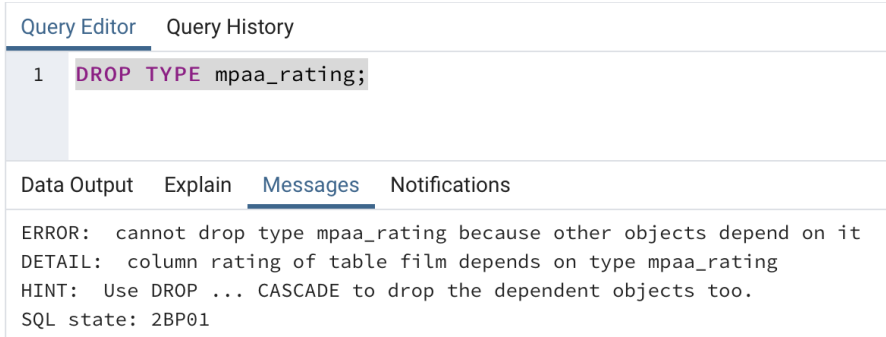
```

Listing 17: SQL-17

What is the response of the RDBMS?

- A. The database returns an error, because the table `film` contains a column on type `mpaa_rating`
- B. The database returns an error, because the syntax of the statement is not correct
- C. The database accepts the schema change, and no changes happen to the database state
- D. The database accepts the additional constraint, but all the tuples in the `film` table are deleted to avoid integrity issues

Solution:



```
Query Editor  Query History
1 DROP TYPE mpaa_rating;

Data Output  Explain  Messages  Notifications
ERROR: cannot drop type mpaa_rating because other objects depend on it
DETAIL: column rating of table film depends on type mpaa_rating
HINT: Use DROP ... CASCADE to drop the dependent objects too.
SQL state: 2BP01
```

Question 36

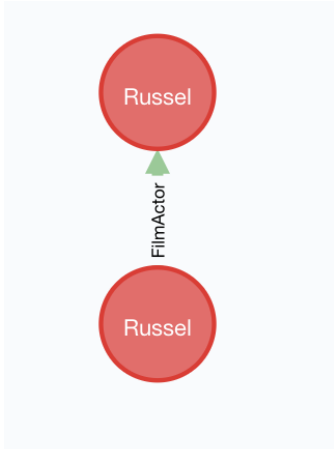
Consider the CYPHER statement below. How many new `Actor` nodes and `FilmActor` relationships are created in the database?

```
1 CREATE p =(:Actor { first_name:'Jack', last_name:'Russel' })
2       -[:FilmActor]->(:Actor {first_name:'Jack', last_name:'Russel'})
3 RETURN p
```

Listing 18: Cypher-1

- A. 2 nodes and 1 relationship
- B. 1 node and 1 relationship
- C. 2 nodes and 2 relationships
- D. The query is syntactically incorrect

Solution:



```
graph BT
  Russel1((Russel)) -- FilmActor --> Russel2((Russel))
```

Question 37

Consider the CYPHER statement below. How many new Actor nodes and FilmActor relationships are created in the database?

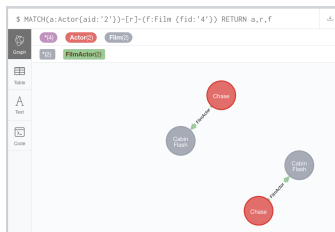
```

1 CREATE p =(:Actor { aid:'2', first_name:'Ed', last_name:'Chase' })-[:FilmActor]->
2   (:Film {fid: '4', name: 'CabinFlash', release_year:2002, rating:'PG-13'})
3 RETURN p

```

Listing 19: Cypher-2

- A. 2 nodes and 1 relationship
- B. 1 node and 1 relationship**
- C. Only 1 relationships
- D. No nodes and no relationships

Solution:**Question 38**

Consider the CYPHER statement below. How many new Actor nodes and FilmActor relationships are created in the database?

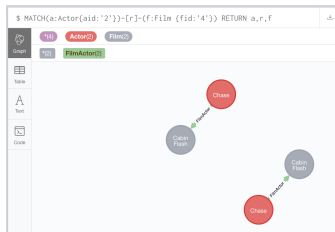
```

1 CREATE p =(:Actor { first_name:'Jack', last_name:'Russel' })
2   -[:FilmActor]->(:Actor {first_name:'Jack', last_name:'Russel'})
3 RETURN p

```

Listing 20: Cypher-3

- A. 2 nodes and 1 relationship**
- B. 1 node and 1 relationship
- C. 2 nodes and 2 relationships
- D. The query is syntactically incorrect

Solution:

0.2 Simple Queries

Question 39

How many results the query below produces?

```
1 MATCH (a:Actor)-[fa:FilmActor]->(f:Film)
2 RETURN a.last_name;
```

Listing 21: Cypher-4

A. 23

B. 9

C. 8

D. 15

a.last_name

"Swank"

"Johansson"

"Lollobrigida"

"Guinness"

"Swank"

"Motel"

"Chase"

"Johansson"

"Lollobrigida"

"Davis"

"Guinness"

"Nicholson"

"Guinness"

"Swank"

"Chase"

"Motel"

Solution: Started streaming 23 records after 1 ms and completed after 2 ms.

Question 40

How many results the query below results?

```
1 MATCH (c:Category)-[fa:FilmCategory]->(f:Film)
2 RETURN c.name;
```

Listing 22: Cypher-5

A. 0

B. 5

C. 16

D. 13

\$ MATCH (c:Category)-[fa:FilmCategory]->(f:Film) RETURN c.name;

Table (no changes, no records)

Code

Solution:

Completed after 3 ms.

Question 41

Consider the following queries.

```
1 MATCH (n:Film)-[*..2]-(m:Film)
2 WHERE n.fid = "2"
3 RETURN m
```

Listing 23: Cypher-6

```
1 MATCH (n:Film)-[*..3]-(m:Film)
2 WHERE n.fid = "2"
3 RETURN m
```

Listing 24: Cypher-7

Which of the following answer correctly describes the output of the two queries?

- A. The two queries return the same number of results.
- B. The query in Listing 23 returns more results than the query in Listing 24
- C. The query in Listing 24 returns more results than the query in Listing 23
- D. Both queries are syntactically incorrect.

Solution: Query Listing 25

m
{ "fid": "1", "length": 86, "rating": "PG", "name": "Academy Dinosaur", "release_year": 2006 }
{ "fid": "5", "rating": "NC-17", "name": "Chicago North", "release_year": 2006, "length": 185 }
{ "fid": "4", "rating": "PG-13", "name": "Cabin Flash", "release_year": 2002, "length": 151 }
{ "fid": "7", "length": 141, "rating": "PG", "name": "Dynamite Tarzan", "release_year": 2009 }
{ "fid": "8", "rating": "PG-13", "name": "Elf Murder", "release_year": 2008, "length": 91 }
{ "fid": "8", "rating": "PG-13", "name": "Elf Murder", "release_year": 2008, "length": 91 }
{ "fid": "4", "rating": "PG-13", "name": "Cabin Flash", "release_year": 2002, "length": 151 }
{ "fid": "6", "rating": "PG-13", "name": "Darkness War", "length": 99, "release_year": 2008 }
{ "fid": "4", "rating": "PG-13", "name": "Cabin Flash", "release_year": 2002, "length": 151 }
{ "fid": "5", "rating": "NC-17", "name": "Chicago North", "release_year": 2006, "length": 185 }
{ "fid": "1", "length": 86, "rating": "PG", "name": "Academy Dinosaur", "release_year": 2006 }
{ "fid": "8", "rating": "PG-13", "name": "Elf Murder", "release_year": 2008, "length": 91 }

Query Listing 26

"m"
{ "fid": "1", "length": 86, "rating": "PG", "name": "Academy Dinosaur", "release_year": 2006 }
{ "fid": "5", "rating": "NC-17", "name": "Chicago North", "release_year": 2006, "length": 185 }
{ "fid": "4", "rating": "PG-13", "name": "Cabin Flash", "release_year": 2002, "length": 151 }
{ "fid": "7", "length": 141, "rating": "PG", "name": "Dynamite Tarzan", "release_year": 2009 }
{ "fid": "8", "rating": "PG-13", "name": "Elf Murder", "release_year": 2008, "length": 91 }
{ "fid": "8", "rating": "PG-13", "name": "Elf Murder", "release_year": 2008, "length": 91 }
{ "fid": "4", "rating": "PG-13", "name": "Cabin Flash", "release_year": 2002, "length": 151 }
{ "fid": "6", "rating": "PG-13", "name": "Darkness War", "length": 99, "release_year": 2008 }
{ "fid": "4", "rating": "PG-13", "name": "Cabin Flash", "release_year": 2002, "length": 151 }
{ "fid": "5", "rating": "NC-17", "name": "Chicago North", "release_year": 2006, "length": 185 }
{ "fid": "1", "length": 86, "rating": "PG", "name": "Academy Dinosaur", "release_year": 2006 }
{ "fid": "8", "rating": "PG-13", "name": "Elf Murder", "release_year": 2008, "length": 91 }

Question 42

Consider the following queries.

```

1 MATCH (n:Film) -[*..1] -() -[*..1] - (p:Film)
2 WHERE n.fid = "2"
3 RETURN p

```

Listing 25: Cypher-8

```

1 MATCH (n:Film) -[*..2] -() -[*..1] - (p:Film)
2 WHERE n.fid = "2"
3 RETURN p

```

Listing 26: Cypher-9

Which of the following answer correctly describes the output of the two queries?

- A. The two queries return the same number of results.**
- B. The query in Listing 25 returns more results than the query in Listing 26
- C. The query in Listing 26 returns more results than the query in Listing 25
- D. Both queries are syntactically incorrect.

Solution: Query Listing 25

"p"
{ "fid": "1", "length": 86, "rating": "PG", "name": "Academy Dinosaur", "release_year": 2006 }
{ "fid": "5", "rating": "NC-17", "name": "Chicago North", "release_year": 2006, "length": 185 }
{ "fid": "4", "rating": "PG-13", "name": "Cabin Flash", "release_year": 2002, "length": 151 }
{ "fid": "7", "length": 141, "rating": "PG", "name": "Dynamite Tarzan", "release_year": 2009 }
{ "fid": "8", "rating": "PG-13", "name": "Elf Murder", "release_year": 2008, "length": 91 }
{ "fid": "8", "rating": "PG-13", "name": "Elf Murder", "release_year": 2008, "length": 91 }
{ "fid": "4", "rating": "PG-13", "name": "Cabin Flash", "release_year": 2002, "length": 151 }
{ "fid": "6", "rating": "PG-13", "name": "Darkness War", "length": 99, "release_year": 2008 }
{ "fid": "4", "rating": "PG-13", "name": "Cabin Flash", "release_year": 2002, "length": 151 }
{ "fid": "5", "rating": "NC-17", "name": "Chicago North", "release_year": 2006, "length": 185 }
{ "fid": "1", "length": 86, "rating": "PG", "name": "Academy Dinosaur", "release_year": 2006 }
{ "fid": "8", "rating": "PG-13", "name": "Elf Murder", "release_year": 2008, "length": 91 }

Query Listing 26

"p"
{ "fid": "1", "length": 86, "rating": "PG", "name": "Academy Dinosaur", "release_year": 2006 }
{ "fid": "5", "rating": "NC-17", "name": "Chicago North", "release_year": 2006, "length": 185 }
{ "fid": "4", "rating": "PG-13", "name": "Cabin Flash", "release_year": 2002, "length": 151 }
{ "fid": "7", "length": 141, "rating": "PG", "name": "Dynamite Tarzan", "release_year": 2009 }
{ "fid": "8", "rating": "PG-13", "name": "Elf Murder", "release_year": 2008, "length": 91 }
{ "fid": "8", "rating": "PG-13", "name": "Elf Murder", "release_year": 2008, "length": 91 }
{ "fid": "4", "rating": "PG-13", "name": "Cabin Flash", "release_year": 2002, "length": 151 }
{ "fid": "6", "rating": "PG-13", "name": "Darkness War", "length": 99, "release_year": 2008 }
{ "fid": "4", "rating": "PG-13", "name": "Cabin Flash", "release_year": 2002, "length": 151 }
{ "fid": "5", "rating": "NC-17", "name": "Chicago North", "release_year": 2006, "length": 185 }
{ "fid": "1", "length": 86, "rating": "PG", "name": "Academy Dinosaur", "release_year": 2006 }
{ "fid": "8", "rating": "PG-13", "name": "Elf Murder", "release_year": 2008, "length": 91 }

Question 43

Consider the following queries.

```

1 MATCH (n:Film)
2   WHERE n.fid = "1"
3   RETURN size((n)--()) AS count

```

Listing 27: Cypher-10

```

1 MATCH (n:Film)
2   WHERE n.fid = "1"
3   RETURN size((n)<-->()) AS count

```

Listing 28: Cypher-11

Which of the following answer correctly describes the output of the two queries?

- A. The query in Listing 28 is syntactically incorrect
- B. The query in Listing 27 returns: **count: 6**, while the query in Listing 28 returns: **count: 5**
- C. The query in Listing 28 returns: **count: 3**, while the query in Listing 28 returns: **count: 6**
- D. The two queries return the same result: count: 6**

Solution: Query Listing 27

"count"
6

Query Listing 28

"count"
6

Question 44

Consider the following queries.

```

1 MATCH (n:Film)
2   WHERE n.fid = "1"
3   RETURN size((n)--()) AS count

```

Listing 29: Cypher-12

```
1 MATCH (n:Film)-[]-()
2 WHERE n.fid = "1"
3 RETURN size((n)--()) AS count
```

Listing 30: Cypher-13

Which of the following answer correctly describes the output of the two queries?

- A. Query 2 is syntactically incorrect
- B. Query 1 is syntactically incorrect
- C. Query 1 returns 1 result, while Query 2 returns 6 results.**
- D. Both queries return the same number of results.

Solution: Query Listing 29

"count"
6

"count"
6
6
6
6
6
6

Query Listing 30

Appendix A - Film Relational Database Schema

The Film database contains data about movies, their categories, and performing actors.

```

1
2 CREATE TYPE db.mpaa_rating AS ENUM ('G', 'PG', 'PG-13', 'R', 'NC-17');
3
4 CREATE DOMAIN db.year AS integer;
5
6 ALTER DOMAIN db.year ADD CONSTRAINT year_check
7     CHECK (VALUE >= 1901 AND VALUE <= 2155);
8
9
10 CREATE TABLE db.actor
11 (
12     actor_id integer NOT NULL,
13     first_name character varying(45) NOT NULL,
14     last_name character varying(45) NOT NULL,
15     CONSTRAINT actor_pkey PRIMARY KEY (actor_id)
16 )
17
18
19 CREATE TABLE db.category
20 (
21     category_id integer NOT NULL,
22     name character varying(25) NOT NULL,
23     CONSTRAINT category_pkey PRIMARY KEY (category_id)
24 )
25
26 CREATE TABLE db.film
27 (
28     film_id integer NOT NULL,
29     title character varying(255) NOT NULL,
30     release_year year,
31     length smallint,
32     rating mpaa_rating DEFAULT 'G',
33     CONSTRAINT film_pkey PRIMARY KEY (film_id)
34 )
35
36 CREATE TABLE db.film_actor
37 (
38     actor_id smallint NOT NULL,
39     film_id smallint NOT NULL,
40     CONSTRAINT film_actor_pkey PRIMARY KEY (actor_id, film_id),
41     CONSTRAINT film_actor_actor_id_fkey FOREIGN KEY (actor_id)
42         REFERENCES db.actor (actor_id) MATCH SIMPLE
43         ON UPDATE CASCADE
44         ON DELETE RESTRICT,
45     CONSTRAINT film_actor_film_id_fkey FOREIGN KEY (film_id)
46         REFERENCES db.film (film_id) MATCH SIMPLE
47         ON UPDATE CASCADE
48         ON DELETE RESTRICT
49 )
50
51 CREATE TABLE db.film_category
52 (
53     film_id smallint NOT NULL,
54     category_id smallint NOT NULL,
55     CONSTRAINT film_category_pkey PRIMARY KEY (film_id, category_id),
56     CONSTRAINT film_category_category_id_fkey FOREIGN KEY (category_id)
57         REFERENCES db.category (category_id) MATCH SIMPLE
58         ON UPDATE CASCADE
59         ON DELETE RESTRICT,
60     CONSTRAINT film_category_film_id_fkey FOREIGN KEY (film_id)
61         REFERENCES db.film (film_id) MATCH SIMPLE
62         ON UPDATE CASCADE
63         ON DELETE RESTRICT
64 )

```


Appendix B - Film Relational Database Instance

category

	category_id [PK] integer	name character varying (25)
1	1	Action
2	2	Animation
3	3	Children
4	4	Classic
5	5	Comedy




film

	film_id [PK] integer	title character varying (255)	release_year integer	length smallint	rating mpaa_rating
1	1	Academy Dinosaur	2006	86	PG
2	2	Baked Cleopatra	2006	86	PG
3	3	Beethoven Exorcist	2012	182	G
4	4	Cabin Flash	2002	151	PG-13
5	5	Chicago North	2006	185	NC-17
6	6	Darkness War	2008	99	PG-13
7	7	Dynamite Tarzan	2009	141	PG
8	8	Elf Murder	2008	91	PG-13




actor

	actor_id [PK] integer	first_name character varying (45)	last_name character varying (45)
1	1	Penelope	Guinness
2	2	Ed	Chase
3	3	Jennifer	Davis
4	4	Johnny	Lollobrigida
5	5	Bette	Nicholson
6	6	Grace	Mostel
7	7	Matthew	Johansson
8	8	Joe	Swank
9	9	Christian	Gable

film_category

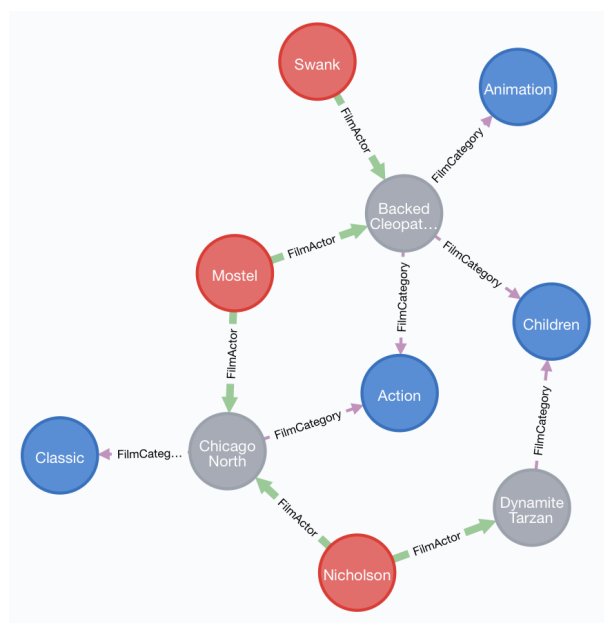
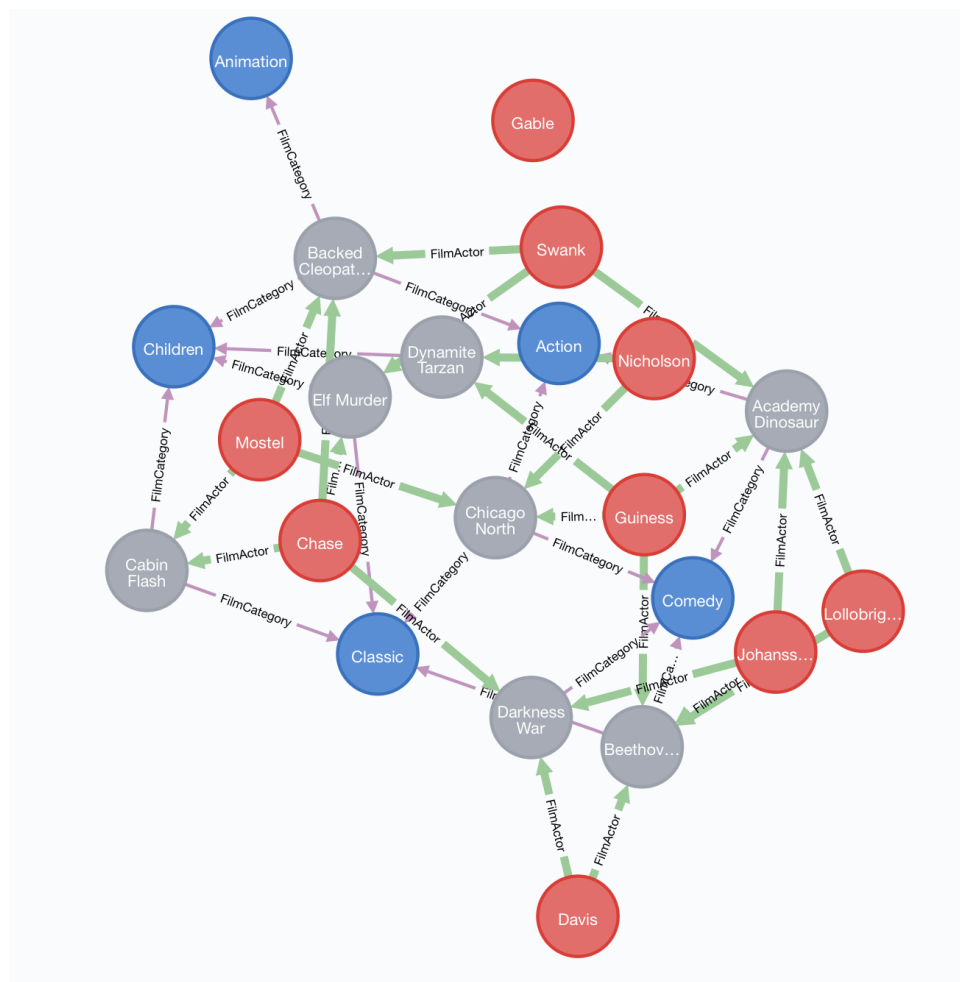
	 film_id [PK] smallint 	category_id [PK] smallint 
1	1	1
2	1	5
3	2	1
4	2	2
5	2	3
6	3	5
7	3	4
8	4	4
9	4	3
10	5	1
11	5	4
12	6	5
13	7	3
14	8	3
15	8	4
16	5	5

film_actor

	 actor_id [PK] smallint 	film_id [PK] smallint 
1	1	1
2	1	3
3	1	5
4	1	7
5	2	2
6	2	4
7	2	6
8	2	8
9	3	3
10	3	6
11	4	1
12	4	3
13	5	5
14	5	7
15	6	2
16	6	4
17	6	5
18	7	1
19	7	3
20	7	6
21	8	1
22	8	2
23	8	8

Appendix C - Film Neo4J Database

Here we provide examples for the Neo4J version of the `film` database.



The Neo4J database contains 3 node labels:

- **category**

Category <id>: 66 cid: 4 name: Classic

- **film**

Film <id>: 42 fid: 8 length: 91 name: Elf Murder rating: PG-13 release_year: 2008

- **actor**

Actor <id>: 24 aid: 5 first_name: Bette last_name: Nicholson

The database contains also 2 relationship types:

- **FilmCategory**: this relationship models the relationship between a film and a category, as contained in the `film_category` relational table

FilmCategory <id>: 41

- **FilmActor**: this relationship models the relationship between an actor and a movie, as contained in the `film_actor` relational table

FilmActor <id>: 30

[Final Page]