COMS W 4111-002
W4111 - Introduction to Databases
Section 003/V03, Fall 2022
Take Home Final

Exam Instructions

- We will publish instructions on Ed.

Environment Setup and Test

MySQL

- Replace `root` and `dbuserdbuser` for the correct values for you MySQL instance from previous homework assignments and exams.
- You will need the sample database that comes with the recommended textbook to execute the setup test.
  - You should have already installed the database because you need for previous assignments.
  - I named my database

In [1]:
```
%load_ext sql
```

In [2]:
```
%sql mysql+pymysql://root:dbuserbdbuser@localhost
```

In [3]:
```
%sql select * from db_book.student
```

```
 * mysql+pymysql://root:***@localhost
0 rows affected.
```

Out[3]:

ID	name	dept_name	tot_cred

Neo4j

- Please set the values for your Neo4j database below.
- Make sure that your database is active. If you have not used it for a while, you need to log in through the website and restart the database.

```
```

In [16]:
```
neo4j_url = "neo4j+s://a8f83b1e.databases.neo4j.io"
neo4j_user = "neo4j"
neo4j_password = '9UZ5prbpDjmXRwKwpvWau4mq1-qAX_NOBGxw-J59kW4'
```

In [7]:
```
from py2neo import Graph
```

In [9]:
```
def t1():
    graph = Graph(neo4j_url, auth=(neo4j_user, neo4j_password))
    q = "match (r:Person) where r.name='Tom Hanks' return r"
    res = graph.run(q)

    for r in res:
        print(r)
```

- Please rerun the following cell.

In [10]:
```
t1()
```

```
Node('Person', born=1956, name='Tom Hanks')
```

MongoDB

- Please set your URL for MongoDB Atlas and make sure that your cluster is not suspended.

In [23]:
```
mongodb_url = "mongodb+srv://honghao_liu:Zi1MwBE3qqt0eOGE@cluster0.wejegit.mongodb.n
```

In [24]:
```
import pymongo
```

In [25]:
```
def connect():
    client = pymongo.MongoClient(
        mongodb_url
    )
    return client


def t_connect():
    c = connect()
    print("Databases = ", list(c.list_database_names()))
```

In [26]:
```
#
# Note, you list of local databases will be different. The values do not matter.
#
t_connect()
```

```
Databases =  ['hw4', 'sample_airbnb', 'sample_analytics', 'sample_geospatial', 'samp
le_guides', 'sample_mflix', 'sample_restaurants', 'sample_supplies', 'sample_trainin
g', 'sample_weatherdata', 'testdb', 'admin', 'local']
```

Written Questions — General Knowledge

- The written questions require a short, succinct answer.
- Remember, "If you can't explain it simply, you don't understand it well enough."

- Some questions will research using the web, lecture slides, etc. You cannot cut and paste from sources. Your answer must show that you read the material and understand the concept.
- If you use a source other than lecture material, please provide a URL to the source(s) you read.

G1

Question: List at least two reasons why database systems support data manipulation using a declarative query language such as SQL, instead of just providing a library of C or C + + functions to carry out data manipulation.

Answer:

Enter answer.

- Declarative languages are easier to learn and use.
- Declarative specifications make it easier for database systems to choose the appropriate execution technology.

G2

Question: List four significant differences between:

- Processing data by writing programs that manipulate files.
- Using a database management system and query language.

Answer:

Enter answer.

- Data redundancy and inconsistency: Redundancy is the concept of repetition of data i.e. each data may have more than a single copy. The file system cannot control the redundancy of data as each user defines and maintains the needed files for a specific application to run.Whereas DBMS controls redundancy by maintaining a single repository of data that is defined once and is accessed by many users. As there is no or less redundancy, data remains consistent.
- Data concurrency: Concurrent access to data means more than one user is accessing the same data at the same time. Anomalies occur when changes made by one user get lost because of changes made by another user. The file system does not provide any procedure to stop anomalies. Whereas DBMS provides a locking system to stop anomalies to occur.
- Data searching: For every search operation performed on the file system, a different application program has to be written. While DBMS provides inbuilt searching operations. The user only has to write a small query to retrieve data from the database.
- Data integrity: There may be cases when some constraints need to be applied to the data before inserting it into the database. The file system does not provide any

procedure to check these constraints automatically. Whereas DBMS maintains data integrity by enforcing user-defined constraints on data by itself.

G3

Question: List five responsibilities (functionality provided) of a database-management system. For each responsibility, explain the potential problems that would occur with the functionality.

Answer:

Enter answer.

- Interaction with the File Manager:If there is no file manager interaction then nothing stored in the files can be retrieved.
- Integrity Enforcement:Consistency constraints may not be satisfied.
- Security Enforcement:Unauthorized users may access the database, or users authorized to access part of the database may be able to access parts of the database for which they lack authority.
- Backup and Recovery:Data could be lost permanently, rather than at least being available in a consistent state that existed prior to a failure.
- Concurrency Control:Consistency constraints may be violated despite proper integrity enforcement in each transaction.

G4

Question: We all use SSOL to choose and register for classes. Another option would be to have a single Google sheet (shared spreadsheet) that we all use to register for classes. What are problems with using a shared spreadsheet?

Answer:

Enter answer.

- Security:Anybody can register for classes even without authority.
- Integrity:The id of every student can be changed anyway.
- Recovery:The lost data cannot be recoverd.
- Concurrency: The conflict will occur when user modify the data at the same time.

G5

Question: NoSQL databases have become increasingly popular for supporting applications. List 3 benefits of or reasons for using NoSQL databases versus SQL/relational databases. List 3 benefits of relational databases versus NoSQL databases.

Answer:

Enter answer. SQL Database benefits:

- SQL databases use a powerful language "Structured Query Language" to define and manipulate the data.
- SQL databases are best suited for complex queries.
- SQL databases enforce constraints, such as primary keys and foreign keys, to ensure the integrity of the data. This makes them well-suited for storing and managing data that needs to be accurate and consistent.

NoSQL databases benefits:

- NoSQL databases are best suited for hierarchical data storage.
- NoSQL databases do not have a fixed schema, which means that they can handle a wide variety of data types and structures. This makes them well-suited for storing unstructured or semi-structured data, such as documents or social media posts.
- NoSQL handle large volumes of data at high speed with a scale-out architecture

Relational Model

R1

Question: A column in a relation (table) has a *type*. Consider implementing a `date` as `CHAR(10)` in the format `YYYY-MM-DD.` The lecture material states that attributes (column values) come from a *domain*. Using `date` explain the differenc between a *domain* and a *type*.

Answer:

Enter answer. Domain is the values the attribute can take. Type is used to limit the attributes value type. For example, a "int" type can only take numbers and a "varchar" type can only take characters. IN this example, the type of date is ten characters because the type for this attributes is CHAR(10). And the ten characters must organized as "YYYY-MM-DD" format because the value domain is in that format.

R2

Question: The domain for a relation (table) attribute (column) should be *atomic*. Why?

Answer:

Enter answer. Atomic means that the relation column cannot be divided into small pieces. Domains restrict the form of attributes. Therefore, the domain must be atomic to ensure integerity.

R3

Question: "In the US Postal System, a delivery point is a specific set of digits between 00 and 99 assigned to every address. When combined with the ZIP + 4 code, the delivery point

provides a unique identifier for every deliverable address served by the United States Postal Service."

The lecture 2 slides provide a notation for representing a relation's schema. Assume we want to define a relation for US mailing addresses, and that the columns are:

- Zip code
- +4 code
- delivery_point
- address_line_1
- address_line_2
- city
- state

Use the notation to define the schema for an address. A simple example of an address's column values might be:

- Zip code: 10027
- +4 code: 6623
- delivery_point: 99
- address_line_1: 520 W 120th St
- address_line_2: Room 402
- city: New York
- state: NY

Answer:

The defination of an address should be:

CREATE TABLE Address(

Zip code int(5),

{+4 code} int(4),

{delivery_point} int(2),

address_line_1 varchar(128),

address_line_2 varchar(128),

city varchar(10),

state varchar(5)

)

R4

Note: Use the RelaX calculator and the schema associated with the recommended textbook to answer this question. Your answer should contain:

- The text for the query.
- An image showing the query execution and result.

An example of the format is:

Query

```
σ capacity >= 50 (classroom)
```

Execution

Question: Translate the following SQL statement into an equivalent relational algebra statement.

```
select
    *
from
    instructor
where
    dept_name in (select dept_name from department where budget >=
100000)
```

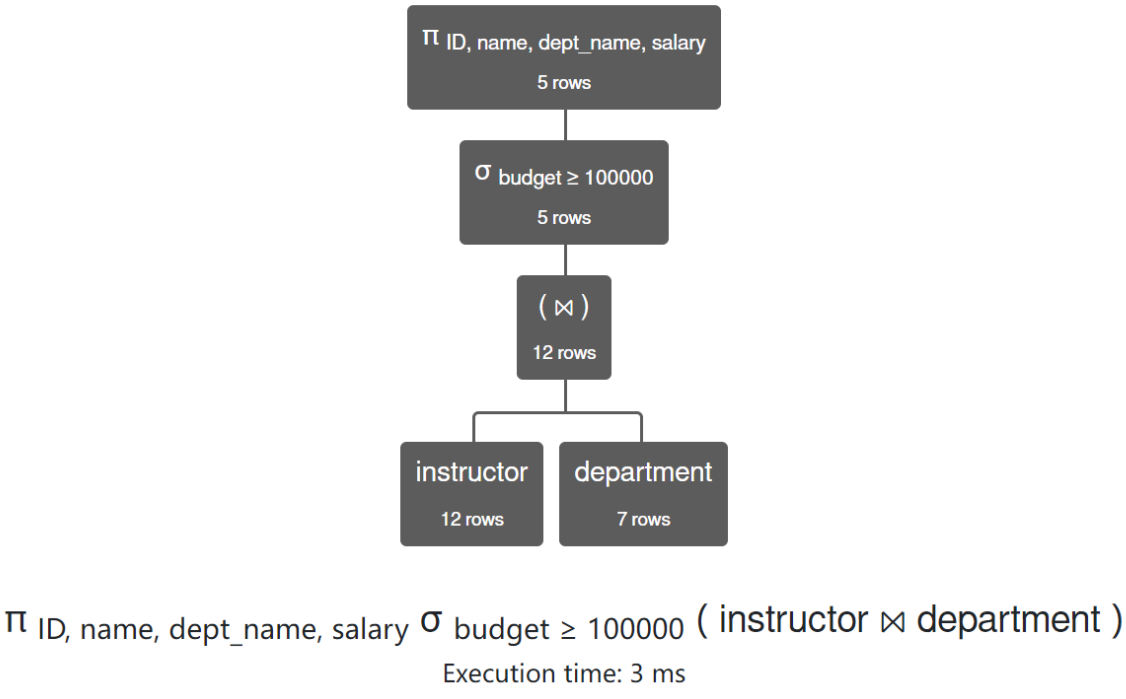
Answer:

Enter answer.

π ID,name,dept_name,salary σ budget≥100000(instructor⋈department)

In [117…
```
from IPython.display import Image
Image(filename = './R4_query.png')
```

Out[117]:



In [118…
```
Image(filename = './R4_result.png')
```

Out[118]:

instructor.ID	instructor.name	instructor.dept_name	instructor.salary
10101	'Srinivasan'	'Comp. Sci.'	65000
12121	'Wu'	'Finance'	90000
45565	'Katz'	'Comp. Sci.'	75000
76543	'Singh'	'Finance'	80000
83821	'Brandt'	'Comp. Sci.'	92000

R5

Use the same format to answer this question.

Question

Use the following query to compute a new table.

```
section_and_time =
    π course_id, sec_id, semester, year,
    day, start_hr, start_min (section ⋈ time_slot)
```

Using only section_and_time, write a relational algebra expression that returns a relation of overlapping courses of the form

`(course_id_1, sec_id_1, semester_1, year_1, course_id_2, sec_id_2, semester_2, year_2)` .

Your table cannot container duplicates. For example, a result containing

```
(BIO-101, 1, fall, 2022, MATH-101, 2, fall, 2022)
(MATH-101, 2, fall, 2022, BIO-101, 1, fall, 2022)
```

is incorrect.

Answer:

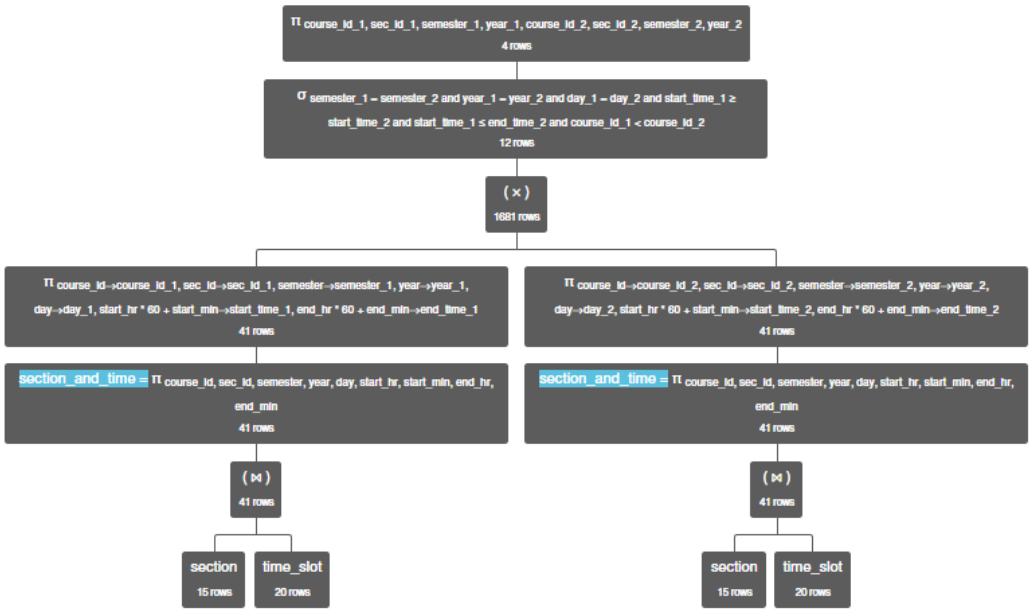
Query π course_id_1, sec_id_1, semester_1, year_1, course_id_2, sec_id_2, semester_2, year_2 (σ semester_1=semester_2∧year_1=year_2∧day_1=day_2∧start_time_1≥start_time_2∧start_time_1 ((π course_id→course_id_1, sec_id→sec_id_1, semester→semester_1, year→year_1, day→day_1, start_hr*60+start_min→start_time_1* ,*end_hr*60+end_min→end_time_1(section_and_time)) × (π course_id→course_id_2,

sec_id→sec_id_2, semester→semester_2, year→year_2, day→day_2,
start_hr*60+start_min→start_time_2,end_hr*60+end_min→end_time_2(section_and_time))))

Execution

In [126…
```
Image(filename = './R5_query.png')
```

Out[126]:



In [127…
```
Image(filename = './R5_result.png')
```

Out[127]:

course_id_1	sec_id_1	semester_1	year_1	course_id_2	sec_id_2	semester_2	year_2
'CS-315'	1	'Spring'	2010	'MU-199'	1	'Spring'	2010
'CS-319'	1	'Spring'	2010	'FIN-201'	1	'Spring'	2010
'CS-319'	2	'Spring'	2010	'HIS-351'	1	'Spring'	2010
'CS-347'	1	'Fall'	2009	'PHY-101'	1	'Fall'	2009

SQL

- You will use the Classic Models tutorial database, which you should have already loaded into MySQL.

S1

Question: Create a view `employee_customer_sales` with the following information:

- `employeeNumber`
- `employeeLastname`

- `employeeFirstName`
- `customerNumber`
- `customerName`
- `revenue`

- The employee information is for the employee that is the `customer.customerRepEmployeeNumber`.

- `revenue` is the total revenue over all of the customer's orders.
  - The revenue for an `order` is `priceEach*quantityOrdered` for each `orderdetails` in the order.

Answer:

In [5]:
```
%sql USE classicmodels
```

```
 * mysql+pymysql://root:***@localhost
0 rows affected.
```

Out[5]:
```
[]
```

In [77]:
```
%%sql
CREATE VIEW employee_customer_sales AS
 SELECT
    employeeNumber,
    lastName as employeeLastName,
    firstName as employeeFirstName,
    customers.customerNumber,
    customerName,
    sum(priceEach*quantityOrdered) as revenue
 FROM
    orderdetails

        INNER JOIN
         orders
        ON orderdetails.orderNumber = orders.orderNumber

        INNER JOIN
         customers
        ON orders.customerNumber = customers.customerNumber

        INNER JOIN
         employees
        ON customers.salesRepEmployeeNumber = employees.employeeNumber

 GROUP BY customers.customerNumber
```

```
 * mysql+pymysql://root:***@localhost
0 rows affected.
```

Out[77]:
```
[]
```

Test Answer:

In [78]:
```
%sql select * from employee_customer_sales;
```

```
 * mysql+pymysql://root:***@localhost
98 rows affected.
```

Out[78]:

employeeNumber	employeeLastName	employeeFirstName	customerNumber	customerName
1370	Hernandez	Gerard	103	Atelier graphique
1166	Thompson	Leslie	112	Signal Gift Stores
1611	Fixter	Andy	114	Australian Collectors, Co.
1370	Hernandez	Gerard	119	La Rochelle Gifts
1504	Jones	Barry	121	Baane Mini Imports
1165	Jennings	Leslie	124	Mini Gifts Distributors Ltd.
1504	Jones	Barry	128	Blauer See Auto, Co.
1165	Jennings	Leslie	129	Mini Wheels Co.
1323	Vanauf	George	131	Land of Toys Inc.
1370	Hernandez	Gerard	141	Euro+ Shopping Channel
1504	Jones	Barry	144	Volvo Model Replicas, Co
1401	Castillo	Pamela	145	Danish Wholesale Imports
1337	Bondur	Loui	146	Saveley & Henriot, Co.
1621	Nishi	Mami	148	Dragon Souveniers, Ltd.
1286	Tseng	Foon Yue	151	Muscle Machine Inc
1216	Patterson	Steve	157	Diecast Classics Inc.
1165	Jennings	Leslie	161	Technics Stores Inc.
1612	Marsh	Peter	166	Handji Gifts& Co
1504	Jones	Barry	167	Herkku Gifts
1370	Hernandez	Gerard	171	Daedalus Designs Imports
1337	Bondur	Loui	172	La Corne D'abondance, Co.
1188	Firrelli	Julie	173	Cambridge Collectables Co.
1323	Vanauf	George	175	Gift Depot Inc.
1621	Nishi	Mami	177	Osaka Souveniers Co.
1286	Tseng	Foon Yue	181	Vitachrome Inc.
1501	Bott	Larry	186	Toys of Finland, Co.
1501	Bott	Larry	187	AV Stores, Co.
1504	Jones	Barry	189	Clover Collections, Co.
1216	Patterson	Steve	198	Auto-Moto Classics Inc.

1501	Bott	Larry	201	UK Collectables, Ltd.
1323	Vanauf	George	202	Canadian Gift Exchange Network
1188	Firrelli	Julie	204	Online Mini Collectables
1166	Thompson	Leslie	205	Toys4GrownUps.com
1370	Hernandez	Gerard	209	Mini Caravy
1621	Nishi	Mami	211	King Kong Collectables, Co.
1702	Gerard	Martin	216	Enaco Distributors
1166	Thompson	Leslie	219	Boards & Toys Co.
1401	Castillo	Pamela	227	Heintze Collectables
1286	Tseng	Foon Yue	233	Québec Home Shopping Network
1166	Thompson	Leslie	239	Collectable Mini Designs Co.
1501	Bott	Larry	240	giftsbymail.co.uk
1370	Hernandez	Gerard	242	Alpha Cognac
1401	Castillo	Pamela	249	Amica Models & Co.
1337	Bondur	Loui	250	Lyon Souveniers
1370	Hernandez	Gerard	256	Auto Associés & Cie.
1504	Jones	Barry	259	Toms Spezialitäten, Ltd
1323	Vanauf	George	260	Royal Canadian Collectables, Ltd.
1611	Fixter	Andy	276	Anna's Decorations, Ltd
1401	Castillo	Pamela	278	Rovelli Gifts
1611	Fixter	Andy	282	Souveniers And Things Co.
1216	Patterson	Steve	286	Marta's Replicas Co.
1702	Gerard	Martin	298	Vida Sport, Ltd
1504	Jones	Barry	299	Norway Gifts By Mail, Co.
1501	Bott	Larry	311	Oulu Toy Supplies, Inc.
1401	Castillo	Pamela	314	Petit Auto
1323	Vanauf	George	319	Mini Classics
1188	Firrelli	Julie	320	Mini Creations Ltd.
1165	Jennings	Leslie	321	Corporate Gift Ideas Co.
1612	Marsh	Peter	323	Down Under Souveniers, Inc

1501	Bott	Larry	324	Stylish Desk Decors, Co.
1323	Vanauf	George	328	Tekni Collectables Inc.
1611	Fixter	Andy	333	Australian Gift Network, Co
1501	Bott	Larry	334	Suominen Souveniers
1188	Firrelli	Julie	339	Classic Gift Ideas, Inc
1702	Gerard	Martin	344	CAF Imports
1166	Thompson	Leslie	347	Men 'R' US Retailers, Ltd.
1337	Bondur	Loui	350	Marseille Mini Autos
1337	Bondur	Loui	353	Reims Collectables
1612	Marsh	Peter	357	GiftsForHim.com
1216	Patterson	Steve	362	Gifts4AllAges.com
1216	Patterson	Steve	363	Online Diecast Creations Co.
1188	Firrelli	Julie	379	Collectables For Less Inc.
1401	Castillo	Pamela	381	Royale Belge
1401	Castillo	Pamela	382	Salzburg Collectables
1621	Nishi	Mami	385	Cruz & Sons Co.
1401	Castillo	Pamela	386	L'ordine Souveniers
1621	Nishi	Mami	398	Tokyo Collectables, Ltd
1337	Bondur	Loui	406	Auto Canal+ Petit
1612	Marsh	Peter	412	Extreme Desk Decorations, Ltd
1504	Jones	Barry	415	Bavarian Collectables Imports, Co.
1286	Tseng	Foon Yue	424	Classic Legends Inc.
1323	Vanauf	George	447	Gift Ideas Corp.
1504	Jones	Barry	448	Scandinavian Gift Ideas
1165	Jennings	Leslie	450	The Sharp Gifts Warehouse
1401	Castillo	Pamela	452	Mini Auto Werke
1286	Tseng	Foon Yue	455	Super Scale Inc.
1286	Tseng	Foon Yue	456	Microscale Inc.
1702	Gerard	Martin	458	Corrida Auto

				Replicas, Ltd
1216	Patterson	Steve	462	FunGiftIdeas.com
1611	Fixter	Andy	471	Australian Collectables, Ltd
1401	Castillo	Pamela	473	Frau da Collezione
1166	Thompson	Leslie	475	West Coast Collectables Co.
1702	Gerard	Martin	484	Iberia Gift Imports, Corp.
1323	Vanauf	George	486	Motor Mint Distributors Inc.
1165	Jennings	Leslie	487	Signal Collectibles Ltd.
1501	Bott	Larry	489	Double Decker Gift Stores, Ltd
1188	Firrelli	Julie	495	Diecast Collectables
1612	Marsh	Peter	496	Kelly's Gift Shop

S2

Question:

- Below, there is a query that creates a view. Run the query.
- Using the view, write a query that produces a table of the form `(productCode, productName)` for products that no customer in Asia has ordered.
- For this questions purposes, the Asian countries are:
  - Japan
  - Singapore
  - Philipines
  - Hong King
- You must not use a JOIN.

In [30]:
```
#
# Create the view
#
%sql create or replace view orders_all as \
    select * from orders join orderdetails using(orderNumber)
```

```
 * mysql+pymysql://root:***@localhost
0 rows affected.
```

Out[30]:
```
[]
```

Answer:

Because there's an item that hasn't sold, so it's Answer1 if you don't count it, and Answer2 if you count it.

Answer1

In [14]:

```
%%sql
SELECT
 productCode,
 productName
FROM
 products
WHERE
  productCode
IN (
    SELECT
     productCode
    FROM
     orders_all
    WHERE
      productCode
    NOT IN (
            SELECT
             productCode
            FROM
             orders_all
            WHERE
             customerNumber
            IN (
                SELECT
                 customerNumber
                FROM
                 customers
                WHERE
                 country
                IN (
                "Hong Kong","Japan","Singapore","Philippines"
                )
               )
            )
  )
```

```
 * mysql+pymysql://root:***@localhost
14 rows affected.
```

Out[14]:

productCode	productName
S10_1678	1969 Harley Davidson Ultimate Chopper
S10_4757	1972 Alfa Romeo GTA
S12_2823	2002 Suzuki XREO
S18_1342	1937 Lincoln Berline
S18_1367	1936 Mercedes-Benz 500K Special Roadster
S18_2795	1928 Mercedes-Benz SSK
S18_2870	1999 Indy 500 Monte Carlo SS
S18_3029	1999 Yamaha Speed Boat
S18_3320	1917 Maxwell Touring Car
S18_3856	1941 Chevrolet Special Deluxe Cabriolet
S24_2022	1938 Cadillac V-16 Presidential Limousine
S24_2972	1982 Lamborghini Diablo
S24_4258	1936 Chrysler Airflow
S700_3505	The Titanic

Answer2

In [122…

```
%%sql
SELECT
 productCode,
 productName
FROM
 products
WHERE
 productCode
NOT IN(
    SELECT
     orders_all.productCode
    FROM
     orders_all
    WHERE
     orders_all.customerNumber
    IN(
        SELECT
         customerNumber
        FROM
         customers
        WHERE
         country
        IN (
            'Japan', 'Singapore', 'Philippines', 'Hong Kong'
        )
    )
)
```

```
 * mysql+pymysql://root:***@localhost
15 rows affected.
```

Out[122]:

productCode	productName
S10_1678	1969 Harley Davidson Ultimate Chopper
S10_4757	1972 Alfa Romeo GTA
S12_2823	2002 Suzuki XREO
S18_1342	1937 Lincoln Berline
S18_1367	1936 Mercedes-Benz 500K Special Roadster
S18_2795	1928 Mercedes-Benz SSK
S18_2870	1999 Indy 500 Monte Carlo SS
S18_3029	1999 Yamaha Speed Boat
S18_3233	1985 Toyota Supra
S18_3320	1917 Maxwell Touring Car
S18_3856	1941 Chevrolet Special Deluxe Cabriolet
S24_2022	1938 Cadillac V-16 Presidential Limousine
S24_2972	1982 Lamborghini Diablo
S24_4258	1936 Chrysler Airflow
S700_3505	The Titanic

S3

Question:

- Use the `customers` and `orders` for this query.
- Shipping days is the number of days between `orderDate` and `shippedDate.`
- Product a table of the form:
  - `customerNumber`
  - `customerName`
  - `noOfOrders` is the number of orders the customer placed.
  - `averageShippingDays`, which is the average shipping days.
  - `minimumShippingDays`, which is the minimum shipping days.
  - `maximumShippingDays`, which is the maximum shipping days.
- The table should only contain entries where:
  - `noOfOrders >= 3`
  - `averageShippingDays >= 5` or `maximumShippingDays >= 10.`

Answer:

In [7]:
```
%%sql
SELECT
 *
FROM
 (SELECT
   customers.customerNumber,
   customerName,
```

```
    COUNT(*) AS noOfOrders,
    AVG(TIMESTAMPDIFF(DAY, orders.orderDate, orders.shippedDate)) AS averageShipping
    MAX(TIMESTAMPDIFF(DAY, orders.orderDate, orders.shippedDate)) AS maximumShipping
    MIN(TIMESTAMPDIFF(DAY, orders.orderDate, orders.shippedDate)) AS minimumShipping
   FROM
    customers

   INNER JOIN
    orders
   ON customers.customerNumber = orders.customerNumber

   GROUP BY customers.customerNumber
  ) AS temp
WHERE
 noOfOrders >= 3
 AND (averageShippingDays >= 5
      OR maximumShippingDays >= 10)
```

```
 * mysql+pymysql://root:***@localhost
12 rows affected.
```

Out[7]:

customerNumber	customerName	noOfOrders	averageShippingDays	maximumShippingDays
363	Online Diecast Creations Co.	3	5.0000	6
385	Cruz & Sons Co.	3	5.3333	6
148	Dragon Souveniers, Ltd.	5	14.6000	65
198	Auto-Moto Classics Inc.	3	5.6667	6
161	Technics Stores Inc.	4	5.2500	6
205	Toys4GrownUps.com	3	5.3333	6
276	Anna's Decorations, Ltd	4	5.0000	6
462	FunGiftIdeas.com	3	5.0000	6
448	Scandinavian Gift Ideas	3	5.5000	6
328	Tekni Collectables Inc.	3	5.0000	6
209	Mini Caravy	3	5.6667	6
398	Tokyo Collectables, Ltd	4	5.5000	8

Graph Database — Neo4j

- You will use your online/cloud Neo4j database for these problems.
- You must have loaded the Movie sample data.

N1

Question:

- The relationship `REVIEWED` connects a `Person` and `Movie`, and has the properties `rating` and `summary`.
- Write Python code using `py2neo` that produces the following table.

Answer:

In [17]:
```
import pandas as pd
from py2neo import Graph
graph = Graph(neo4j_url, auth=(neo4j_user, neo4j_password))
```

In [19]:
```
query = """
match (p:Person)-[r:REVIEWED]->(m:Movie)
return p.name as reviewer_name, r.rating as rating, r.summary as rating_summary, m.t
"""
result = graph.run(query)
df = pd.DataFrame(result, columns=["reviewer_name", "rating", "rating_summary", "mov
```

In [20]:
```
df
```

Out[20]:

	reviewer_name	rating	rating_summary	movie_title
0	Jessica Thompson	92	You had me at Jerry	Jerry Maguire
1	James Thompson	100	The coolest football movie ever	The Replacements
2	Angela Scope	62	Pretty funny at times	The Replacements
3	Jessica Thompson	65	Silly, but fun	The Replacements
4	Jessica Thompson	45	Slapstick redeemed only by the Robin Williams ...	The Birdcage
5	Jessica Thompson	85	Dark, but compelling	Unforgiven
6	Jessica Thompson	95	An amazing journey	Cloud Atlas
7	Jessica Thompson	68	A solid romp	The Da Vinci Code
8	James Thompson	65	Fun, but a little far fetched	The Da Vinci Code

N2

Question:

- There are relationships `ACTED_IN` and `DIRECTED` between `Person` and `Movie`.
- Write Python code that produces the following table that shows people or both acted in and directed a movie.

In [22]:
```
query = """
match (p:Person)-[:ACTED_IN]->(m:Movie)<-[:DIRECTED]-(p)
return p.name as name, m.title as movie
"""

result = graph.run(query)
df = pd.DataFrame(result, columns=["name", "movie"])
df
```

Out[22]:

	name	movie
0	Tom Hanks	That Thing You Do
1	Clint Eastwood	Unforgiven
2	Danny DeVito	Hoffa

MongoDB

- Run the following code using your Atlas MongoDB.

In [110…
```
import json

client = pymongo.MongoClient(
    mongodb_url
)

with open("./episodes.json") as e_file:
    episodes = json.load(e_file)["episodes"]


for e in episodes:
    e['episodeLink'] = e['episodeLink'].split("/")[2]
    client['w4111_final']['episodes'].insert_one(e)
```

In [111…
```
ratings_df = pd.read_csv("./got_title_ratings.csv")
ratings_info = ratings_df[['tconst', 'averageRating', 'numVotes']]
r_dict = ratings_info.to_dict("records")

for r in r_dict:
    client['w4111_final']['ratings'].insert_one(r)
```

Question:

Write Python code that uses an aggregation pipeline and operations to produce the following table.

In [99]:
```
# Requires the PyMongo package.
# https://api.mongodb.com/python/current
#
# Write the query/aggregation that produces result
```

In [112…
```
db = client.get_database("w4111_final")
result = db.episodes.aggregate([{'$lookup':{
            'from': "ratings",
            'localField': "episodeLink",
            'foreignField': "tconst",
            'as': "rating"
        }
    },
    {
       "$unwind":{
        "path": "$rating"
    },

    },
                                    {
```

```
        "$project": {
            "_id": 0,
            "seasonNum": 1,
            "episodeNum": 1,
            "episodeLink": 1,
            "episodeTitle": 1,
            "avgRating": "$rating.averageRating",
            "numVotes": "$rating.numVotes"
        }
    }
])
```

In [113…
```
info_df = pd.DataFrame(list(result))
info_df = info_df[['seasonNum', 'episodeNum', 'episodeLink', 'episodeTitle', 'avgRat
info_df
```

Out[113]:

	seasonNum	episodeNum	episodeLink	episodeTitle	avgRating	numVotes
0	1	1	tt1480055	Winter Is Coming	8.9	48686
1	1	2	tt1668746	The Kingsroad	8.6	36837
2	1	3	tt1829962	Lord Snow	8.5	34863
3	1	4	tt1829963	Cripples, Bastards, and Broken Things	8.6	33136
4	1	5	tt1829964	The Wolf and the Lion	9.0	34436
...	...	...	...	...	...	...
68	8	2	tt6027908	A Knight of the Seven Kingdoms	7.9	130844
69	8	3	tt6027912	The Long Night	7.5	215995
70	8	4	tt6027914	The Last of the Starks	5.5	165067
71	8	5	tt6027916	The Bells	6.0	192449
72	8	6	tt6027920	The Iron Throne	4.0	248318

73 rows × 6 columns

Data Modeling and Schema Definition

- This is an exciting, interesting problem that involves:
    - Using Crow's Foot Notation
    - Relational approaches to implementing specialization, aggregation, quaternary relations, composite attributes and multi-valued attributes.
    - Foreign keys, check constraints and triggers.
- I did the answer and it took 3 hours to do all the work. My normal rule of thumb is that students require about 15 times as much time as I need to produce an answer.
- I giggled like the Riddler in Batman about how much fun we were going to have working on this question, and then the following happened.

- So, there will not be any data modeling question on the exam. Darn!

Module II Questions

- The questions require brief, written answers.

Q1

Question:

Briefly explain:

- Functional Dependency
- Lossy Decomposition
- Normalization

Answer:

- Functional dependency is a relationship between two attributes, typically between the PK and other non-key attributes within a table. If the value for every instance in the non-key attributes depends on the key value. The key is determinant attributes and the non-key is dependent attributes. Their relationship is function dependency.
- Lossy DEcomposition means when a relation is decomposed into two or more relational schemas, the loss of information is unavoidable when the original relation is retrieved.
- Database normalization is a method in relational database design which helps properly organize data tables. The process aims to create a system that faithfully represents information and relationships without data loss or redundancy.

Q2

Question:

Briefly explain:

- Serializability
- Conflict Serializability
- Deadlock
- Cascading Abort
- Two Phase Locking

Answer:

- Serial schedule means that the transactions bestowed upon it will take place serially, that is, one after the other.
- A schedule is called conflict serializability if after swapping of non-conflicting operations, it can transform into a serial schedule.

- A deadlock is an unwanted situation in which two or more transactions are waiting indefinitely for one another to give up locks.
- Cascading abort can occur when transactions are executed concurrently and one transaction modifies data that is needed by another transaction.
- Two phase locking is a concurrency control protocol that ensures that transactions are executed in a way that is serializable. It does this by dividing the execution of transactions into two phases: a growing phase, where transactions can acquire locks on data, and a shrinking phase, where transactions must release their locks.

Q3

Question:

Briefly explain:

- Logical block addressing, CHS addressing
- RAID-0, RAID-1, RAID-5
- Fixed length records, variable length records.

Answer:

- LBA (Logical Block Addressing)is the process of addressing the sectors on a drive as a single group of logical block numbers. CHS addressing is a way of addressing the blocks on a disk drive using a physical address. LBA allows for accessing larger drives than is usually possible.
- RAID 0 will combine two drives and write data on both of them simultaneously or sequentially, which will help with read and write speeds. RAID 1 will duplicate your data and store a copy on each drive. This is called mirroring, and it ensures you won't lose your files if a drive fails.RAID 5 combines striping and parity for speed and redundancy. If you have at least three hard drives, using RAID 5 will break your data into segments and save those segments across your drives.
- Fixed length records have a fixed number of fields and a fixed length. This means that each record has the same number of fields and each field has the same length. Variable length records have a variable number of fields and a variable length. Fixed length records are more efficient to store and retrieve, but they are less flexible than variable length records.

Q4

Question:

Briefly explain:

- Clustered Index
- Sparse Index
- Covering Index

Answer:

- Clustering index is defined on an ordered data file. The data file is ordered on a non-key field.
- Sparse index records are not created for every search key. An index record here contains a search key and an actual pointer to the data on the disk.
- A covering index is an index that contains all of the columns needed to satisfy a query.

Q5

Question:

Briefly explain:

- Equivalent queries
- Hash Join
- Materialization, Pipelining

Answer:

- For a query result, there are many queries can generate the same result. They are called equivalent queries.
- Hash join is a method for execute join operation. Hash join is used when projections of the joined tables are not already sorted on the join columns.
- Materialization is the process of storing the results of a query in a temporary location, such as a temporary table or a view. Pipelining is a technique used to improve the performance of a database query by allowing multiple operations to be performed in parallel.

In [ ]: