Q1. Data Modeling (10 points)

You are working with a Pokémon database that tracks trainers, the Pokémon they own, and the gyms they have visited. The database includes the following SQL table:

Trainers				
id	**name**	**hometown**	**favorite_type**	**gym_id**
1	Ash	Pallet Town	Electric	3
2	Misty	Cerulean	Water	1
3	Brock	Pewter	Rock	2
4	Erika	Celadon	Grass	NULL

GYMs			
d	**name**	**location**	**gym_type**
1	Cerulean Gym	Cerulean	Water
2	Pewter Gym	Pewter	Rock
3	Vermilion Gym	Vermilion	Electric

Deliverables:

1. Equivalent queries for MongoDB, Redis (if possible), and Neo4j.
    1. **NB: Syntax is not important, do not worry if you forget some detail!**
    2. **You can also describe the query**
2. A brief explanation of your approach for each database.
3. Include any assumption in the corresponding top right box if you need.

Queries

SQL	Mongo	Neo4J (Cypher)	Redis
Query 1: Find trainers whose favorite type is "Water" **SELECT** * **FROM** trainers **WHERE** favorite_type = 'Water';		Assumptions:	
Query 2: Count trainers grouped by their hometown **SELECT** hometown, **COUNT**(*) AS numTrnrs **FROM** trainers **GROUP BY** hometown;		Assumptions:	
Query 3: Retrieve trainers and the gyms they have visited **SELECT** t.name, g.name **FROM** trainers t **LEFT** **JOIN** gyms g **ON** t.gym_id = g.id;		Assumptions:	

SQL	Mongo	Neo4J (Cypher)	Redis

Q2. **Design Problem (10 points)**

You are a data engineer at **RealEstateHub**, a platform that aggregates real estate listings from various sources. Every day, your system collects property listing data from **two REST APIs**, as well as property tax rates and zoning information stored in a **CSV file** uploaded daily. These datasets need to be merged, cleaned, and processed to feed a centralized **Property Analytics Data Warehouse** used by analysts to generate reports.

Instructions

1. **Identify the required tasks:** (HINT: 4 + a including a data loading task)
2. **Draw the pipeline at a conceptual level**:
   - Identify sources and sinks.
   - Identify the necessary transformations.
   - Break the transformations into individual tasks.
3. **Draw the pipeline at a logical level**:
   - Design an Airflow DAG to represent the pipeline.
   - Ensure each node represents a task with minimal responsibilities.
4. **Describe the pipeline at a physical level**:
   - Provide hints on how each task can be implemented, focusing on the operators introduced in class.

Focus on modularity, clear task separation, and error handling throughout the pipeline design.

Q3. Data Wrangling (8):

Pre-process the following table to reach the correct answer to the following SQL query calculating the salary and age average. List the necessary cleaning passages (e.g., noisy/missing data handling) and correct the data on the table.

Query	**Average Shooting**	**Average Assists**
SELECT AVG((ShootsIn * 1.0 / ShootsTotal) * 100) **AS** AVG_Shoot, AVG(Assists) **AS** AVG_Assists **FROM** BasketballStats;	45.6	3.2

PlayerName	**ShootsIn**	**ShootsTotal**	**Assists**
John	50	100	5
NULL	30	60	4
Mary	20	NULL	6
Alex	-10	90	-1

PlayerName	ShootsIn	ShootsTotal	Assists
Lisa	40	80	0
Chris	25	50	NULL
Jordan	30	70	2
NULL	NULL	NULL	NULL
Jamie	0	10	3
Taylor	15	30	-5

Q4. (4 points):

Briefly describe the following methodology for data modelling:

- Entity-Relationship
- Dimensional Modelling
- Data Vault Storage

Q5. (4 points):

- Draw and label a diagram that includes the typical stages of the data engineering lifecycle.
- Write a one-sentence explanation of the goal for each stage.
- If applicable, provide one example tool or framework commonly used for each stage (e.g., those we saw in class).

Q6. (4 points):

Provide 3 differences between OLTP and OLAP