Exercici 4. Comparativa sobre diferents esquemes de base de dades: SCRIPT D'IMPORTACIÓ

```
//Creamos restriccion de año y de individuo
CREATE CONSTRAINT padro_year_unique IF NOT EXISTS
FOR (p:Padro)
REQUIRE p.year IS UNIQUE;
CREATE CONSTRAINT individu_id_unique IF NOT EXISTS
FOR (i:Individu)
REQUIRE i.id IS UNIQUE;

//Creamos individuo y año
LOAD CSV WITH HEADERS FROM 'file:///INDIVIDUAL.csv' AS row
WITH row
WHERE row.Id IS NOT NULL AND row.Year IS NOT NULL
WITH row, ToInteger(row.Year) AS year
MERGE (p:Padro {year: year})
MERGE (i:Individu {id: row.Id})
SET i.nom = row.name,
    i.cognom = row.surname,
    i.segon_cognom = row.second_surname,
    i.year = year

//Convertimos año en una lista encadenada
MATCH (p:Padro)
WITH p ORDER BY p.year
WITH collect(p) AS years
UNWIND range(0, size(years)-2) AS i
WITH years[i] AS from, years[i+1] AS to
MERGE (from)-[:NEXT]->(to);

//Carreguem SAME_AS
LOAD CSV WITH HEADERS FROM 'file:///SAME_AS.csv' AS row
WITH row
WHERE row.Id_A IS NOT NULL AND row.Id_B IS NOT NULL

MATCH (a:Individu {id: row.Id_A})
MATCH (b:Individu {id: row.Id_B})
MERGE (a)-[:SAME_AS]->(b)



//Creem cliques relació
MATCH (n:Individu)
WHERE n.componentId IS NULL
CALL apoc.path.subgraphNodes(n, {relationshipFilter: 'SAME_AS'}) YIELD node
WITH n, collect(node) AS component
WITH component, id(n) AS cid
UNWIND component AS x
SET x.componentId = cid


//Creem nodes de referencia
MATCH (a:Individu)
WITH a.componentId AS group, a,
     size([x IN [a.Id, a.Year, a.name, a.surname, a.second_surname] WHERE x IS NOT
NULL]) AS filled
ORDER BY filled DESC
WITH group, collect(a)[0] AS rep
```

```
SET rep:Representant

//Creem la aresta referencia
MATCH (n:Individu)
MATCH (rep:Representant)
WHERE n.componentId = rep.componentId
MERGE (rep)-[:REPRESENTA]->(n)

//Borramos same as, componentId y la autoaresta
MATCH ()-[r:SAME_AS]-()
DELETE r;

MATCH (n)
REMOVE n.componentId

//importamos familia filtrando malos
LOAD CSV WITH HEADERS FROM 'file:///FAMILIA.csv' AS row
WITH row
WHERE row.ID_1 IS NOT NULL AND row.ID_2 IS NOT NULL
  AND row.Relacio_Harmonitzada IS NOT NULL
  AND NOT row.Relacio_Harmonitzada IN ['null', 'ala', 'al']

MATCH (a:Individu {id: row.ID_1})<-[:REPRESENTA]-(repA:Representant)
MATCH (b:Individu {id: row.ID_2})<-[:REPRESENTA]-(repB:Representant)

MERGE (repA)-[r:FAMILIA]->(repB)
SET r.relacio = row.Relacio_Harmonitzada

//importamos ubicación y habitatges y conexion entre ellos y con los años
LOAD CSV WITH HEADERS FROM 'file:///HABITATGES.csv' AS row
MERGE (u:Ubicacio {carrer: row.Carrer, numero: row.Numero, municipi: row.Municipi})
MERGE (h:Habitatge {id: row.Id_Llar, year: toInteger(row.Any_Padro), municipi:
row.Municipi})
MERGE (h)-[:ESTA_A]->(u)
WITH h, row

MATCH (a:Padro {year: toInteger(row.Any_Padro)})
MERGE (h)-[:DEL_ANY]->(a)

//Conectamos habitatge con los individuos
LOAD CSV WITH HEADERS FROM 'file:///VIU.csv' AS row
WITH row
MATCH (h:Habitatge{id:row.HOUSE_ID, year:toInteger(row.Year), municipi:
row.Location})
MATCH (:Individu {id: row.IND})<-[:REPRESENTA]-(repA:Representant)
MERGE (repA)-[:VIU_EN]->(h)

//Creamos arestas para individuos no conectados
MATCH (i:Individu)<-[:REPRESENTA]-(rep:Representant)
WHERE NOT EXISTS ((rep)-[:VIU_EN]->(:Habitatge))
WITH rep, i.year AS y
MERGE (p:Padro {year: y})
MERGE (rep)-[:NO_UBICAT]->(p)

//Creamos la segunda etiqueta de ubicacio
MATCH (u:Ubicacio)
CALL apoc.create.addLabels(u, [u.municipi]) YIELD node
REMOVE node.municipi
```

```
//Borramos los atributos redundantes
MATCH (h:Habitatge)
REMOVE h.municipi, h.year
```

CONSULTES CYPHER

Model nou:

```
MATCH (rep:Representant)-[:VIU_EN]->(:Habitatge)-[:ESTA_A]->(u:Ubicacio)
WITH rep, size(collect(u)) AS n_ubicacions,  collect(DISTINCT u) AS ubicacions
WHERE size(ubicacions) = 1 AND n_ubicacions > 1
RETURN size(collect(rep)) AS SEDENTARIS
```

Model original:

```
MATCH (p:Person)

OPTIONAL MATCH (p)-[:SAME_AS]-(q:Person)
WITH p, collect(q) + p AS group_nodes
WITH [x IN group_nodes WHERE x IS NOT NULL | x.id] AS ids, group_nodes
WITH apoc.coll.sort(ids)[0] AS group_id, collect(DISTINCT p) AS persons
UNWIND persons AS person
MATCH (person)-[:VIU]->(h:Habitatge)
WITH group_id, collect(DISTINCT h.municipi + '|' + h.carrer + '|' + h.numero) AS adreces
WHERE size(adreces) = 1
RETURN count(*) AS SEDENTARIS
```

Persones que apareixen a dos padrons consecutius:

Model nou:

S'identifiquen els individus que viuen en habitatges associats a anys consecutius, utilitzant la relació `NEXT` per establir la continuïtat temporal.

```
MATCH (p1:Padro {year: $any_padro})-[:NEXT]->(p2:Padro)
MATCH (rep:Representant)-[:VIU_EN]->(:Habitatge)-[:DEL_ANY]->(p1)
WITH p1, p2, collect(DISTINCT rep) AS reps1
MATCH (rep:Representant)-[:VIU_EN]->(:Habitatge)-[:DEL_ANY]->(p2)
WITH reps1, collect(DISTINCT rep) AS reps2
WITH reps1, reps2,
    [r IN reps1 WHERE r IN reps2] AS enAmbdos
RETURN
  size(enAmbdos) AS persistents,
  size(reps1) AS primer_any,
```

```
  toFloat(size(enAmbdos)) / size(reps1) * 100 AS percentatge
```

Model original:

```
MATCH (:Habitatge)
WHERE any_padro > year_actual
WITH year_actual, min(any_padro) AS year_seguent
MATCH (p:Person)-[:VIU]->(h1:Habitatge)
WHERE h1.any_padro = year_actual
WITH DISTINCT p, year_actual, year_seguent
OPTIONAL MATCH (p)-[:SAME_AS]-(q:Person)
WITH p, year_actual, year_seguent, collect(q.id) + p.id AS ids
WITH reduce(min_id = p.id, x IN ids | CASE WHEN x < min_id THEN x ELSE min_id END)
AS group_id, ids, year_actual, year_seguent
WITH group_id, ids, year_actual, year_seguent
WITH collect(DISTINCT group_id) AS grups_originals, year_actual, year_seguent
UNWIND grups_originals AS gid
MATCH (p:Person)-[:SAME_AS*0..]-(p_equiv:Person)
WHERE p.id = gid
WITH DISTINCT gid, collect(DISTINCT p_equiv.id) AS all_ids
UNWIND all_ids AS pid
MATCH (p2:Person {id: pid})-[:VIU]->(h2:Habitatge)
WHERE h2.any_padro = year_seguent
WITH DISTINCT gid AS grups_en_ambdós, count(DISTINCT gid) AS n_ambdós
WITH n_ambdós, count(*) + 0.0 AS total_grups
RETURN
  total_grups AS total_cens_inicial, n_ambdós AS persones_amb_cens_repetit,
  round(n_ambdós * 100.0 / total_grups, 2) AS percentatge;
```