//Order

LOAD CSV WITH HEADERS FROM "file:///olist\_orders\_dataset.csv" AS orders

WITH orders WHERE orders.order\_id IS NOT NULL

CREATE (o:Order{order\_id: orders.order\_id,

customer\_id: orders.customer\_id,

purchase\_timestamp: datetime({ epochMillis: apoc.date.parse(orders.order\_purchase\_timestamp, 'ms', 'YYYY-mm-dd HH:mm:ss') }),

delivery\_date: datetime({ epochMillis: apoc.date.parse(orders.order\_estimated\_delivery\_date, 'ms', 'YYYY-mm-dd HH:mm:ss') })})

//Review DA CONTROLLARE

LOAD CSV WITH HEADERS FROM "file:///olist\_order\_reviews\_dataset.csv" AS reviews

WITH reviews WHERE reviews.review\_id IS NOT NULL

CREATE (r:Review{review\_id: reviews.review\_id,

r.score : reviews.review\_score,

r.comment\_title : CASE trim(reviews.comment\_title) WHEN "" THEN null ELSE reviews.comment\_title END,

r.comment : CASE trim(reviews.comment\_message) WHEN "" THEN null ELSE reviews.comment\_message END,

r.creation\_date : datetime({ epochMillis: apoc.date.parse(reviews.creation\_date, 'ms', 'YYYY-mm-dd HH:mm:ss') }),

r.answer\_timestamp : datetime({ epochMillis: apoc.date.parse(reviews.answer\_timestamp

, 'ms', 'YYYY-mm-dd HH:mm:ss') })

})

//Seller

LOAD CSV WITH HEADERS FROM "file:///olist\_sellers\_dataset.csv" AS sellers

WITH sellers WHERE sellers.seller\_id IS NOT NULL

CREATE (s:Seller{seller\_id: sellers.seller\_id})

MERGE (c:City{name\_state: sellers.seller\_city + "-" + sellers.seller\_state})

MERGE (st:State {code: sellers.seller\_state })

//Customer

LOAD CSV WITH HEADERS FROM "file:///olist\_customers\_dataset.csv" AS customers

WITH customers WHERE customers.customer\_id IS NOT NULL AND customers.customer\_unique\_id IS NOT NULL

CREATE (Customer{customer\_id: customers.customer\_unique\_id})

:auto LOAD CSV WITH HEADERS FROM "file:///olist\_customers\_dataset.csv" AS customers

CALL {

WITH customers

WITH customers WHERE customers.customer\_id IS NOT NULL AND customers.customer\_unique\_id IS NOT NULL

MERGE (ct:City {name\_state: customers.customer\_city + "-" +customers.customer\_state})

MERGE (s:State {code: customers.customer\_state })

} IN TRANSACTIONS OF 500 ROWS

//Item

LOAD CSV WITH HEADERS FROM "file:///olist\_order\_items\_dataset.csv" AS items

WITH items WHERE items.order\_id IS NOT NULL AND items.order\_item\_id IS NOT NULL AND items.seller\_id IS NOT NULL

CREATE (i:Item{item\_id: items.order\_id + "-" + items.order\_item\_id, price: items.price})

//Product

LOAD CSV WITH HEADERS FROM "file:///olist\_products\_dataset.csv" AS products

WITH products WHERE products.product\_id IS NOT NULL

CREATE (p:Product{product\_id: products.product\_id, photos\_qty: products.product\_photos\_qty, weight\_g: products.product\_weight\_g, length\_cm: products.product\_length\_cm, height\_cm: products.product\_height\_cm, width\_cm: products.product\_width\_cm})

MERGE (c:Category{name: products.product\_category\_name})

//Payment

LOAD CSV WITH HEADERS FROM "file:///olist\_order\_payments\_dataset.csv" AS payments

WITH payments WHERE payments.order\_id IS NOT NULL

CREATE (p:Payment{order\_id: payments.order\_id, value: payments.payment\_value, type: payments.payment\_type })

CREATE INDEX order\_customer\_id\_index FOR (n:Order) ON (n.customer\_id)

CREATE INDEX customer\_id\_index FOR (n:Customer) ON (n.customer\_id)

LOAD CSV WITH HEADERS FROM "file:///olist\_customers\_dataset.csv" AS customers\_rel

WITH customers\_rel WHERE customers\_rel.customer\_id IS NOT NULL AND customers\_rel.customer\_unique\_id IS NOT NULL

MATCH (c:Customer{customer\_id: customers\_rel.customer\_unique\_id})

MATCH (o:Order{customer\_id: customers\_rel.customer\_id})

MERGE (c)-[:PLACED]->(o)

MATCH (o:Order)

REMOVE o.customer\_id

CREATE INDEX state\_index FOR(s:State) ON (s.code)

CREATE INDEX city\_index FOR(c:City) ON (c.name\_state)

LOAD CSV WITH HEADERS FROM "file:///olist\_customers\_dataset.csv" AS customers\_rel

WITH customers\_rel WHERE customers\_rel.customer\_id IS NOT NULL AND customers\_rel.customer\_unique\_id IS NOT NULL

MATCH (ct:City {name\_state: customers\_rel.customer\_city + "-" +customers\_rel.customer\_state})

MATCH (st:State {code: customers\_rel.customer\_state })

MERGE (ct)-[:PART\_OF]->(st)

LOAD CSV WITH HEADERS FROM "file:///olist\_order\_reviews\_dataset.csv" AS reviews\_rel

WITH reviews\_rel WHERE reviews\_rel.review\_id IS NOT NULL

MATCH (r:Review{review\_id: reviews\_rel.review\_id})

MATCH (o:Order{order\_id: reviews\_rel.order\_id})

MERGE (o)-[:RECEIVED]->(r);

CREATE INDEX seller\_id\_index FOR (n:Seller) ON (n.seller\_id)

LOAD CSV WITH HEADERS FROM "file:///olist\_sellers\_dataset.csv" AS sellers\_rel

WITH sellers\_rel WHERE sellers\_rel.seller\_id IS NOT NULL

MATCH (s:Seller{seller\_id: sellers\_rel.seller\_id})

MATCH (c:City{name\_state: sellers\_rel.seller\_city + "-" + sellers\_rel.seller\_state})

MATCH (st:State {code: sellers\_rel.seller\_state })

MERGE (c)-[:PART\_OF]->(st)

MERGE (s)-[:HAS\_HEADQUARTERS\_IN]->(c);

LOAD CSV WITH HEADERS FROM "file:///olist\_order\_items\_dataset.csv" AS items\_rel

WITH items\_rel WHERE items\_rel.order\_id IS NOT NULL AND items\_rel.order\_item\_id IS NOT NULL AND items\_rel.seller\_id IS NOT NULL

MATCH (i:Item{item\_id: items\_rel.order\_id + "-" + items\_rel.order\_item\_id})

MATCH (o:Order{order\_id: items\_rel.order\_id})

MERGE (o)-[:COMPOSED\_OF]->(i)

MATCH (s:Seller{seller\_id: items\_rel.seller\_id})

MERGE (i)-[:SOLD\_BY]->(s)

MATCH (p:Product{product\_id: items\_rel.product\_id})

MERGE (i)-[:CORRESPONDS\_TO]->(p);

LOAD CSV WITH HEADERS FROM "file:///olist\_products\_dataset.csv" AS products\_rel

WITH products\_rel WHERE products\_rel.product\_id IS NOT NULL

MATCH (p:Product{product\_id: products\_rel.product\_id})

MATCH (c:Category{name: products\_rel.product\_category\_name})

MERGE (p)-[:BELONGS\_TO]->(c);

LOAD CSV WITH HEADERS FROM "file:///olist\_payments \_dataset.csv" AS payments\_rel

WITH payments\_rel WHERE payments\_rel.order\_id IS NOT NULL

MATCH (o:Order{order\_id: payments\_rel.order\_id})

MATCH (p:Payment{order\_id: payments\_rel.order\_id })

ON MATCH REMOVE p.order\_id

MERGE (o)-[:PAID\_BY]->(p)