CREATE CONSTRAINT ON (n:Wine) ASSERT n.ID is UNIQUE;

CREATE CONSTRAINT ON (n:Variety) ASSERT n.name is UNIQUE;

CREATE CONSTRAINT ON (n:Winery) ASSERT n.name is UNIQUE;

CREATE CONSTRAINT ON (n:Points) ASSERT n.score is UNIQUE;

CREATE CONSTRAINT ON (n:Price) ASSERT n.score is UNIQUE;

CREATE CONSTRAINT ON (n:Region1) ASSERT n.name is UNIQUE;

CREATE CONSTRAINT ON (n:Province) ASSERT n.name is UNIQUE;

CREATE CONSTRAINT ON (n:Country) ASSERT n.name is UNIQUE;

LOAD CSV WITH HEADERS FROM 'file:///title.csv' AS row

CREATE (m:Wine {ID: row.ID, name: row.title, description: row.description });

LOAD CSV WITH HEADERS FROM 'file:///variety.csv' AS row

WITH row WHERE NOT row.variety IS null

MERGE (n:Variety {name: row.variety})

WITH row, n

MATCH(w:Wine {ID: row.ID})

MERGE (w)-[:GRAPE\_TYPE]->(n);

LOAD CSV WITH HEADERS FROM 'file:///winery.csv' AS row

WITH row WHERE NOT row.winery IS null

MERGE (n:Winery {name: row.winery})

WITH row, n

MATCH(w:Wine {ID: row.ID})

MERGE (w)-[:VINTRIFIED\_AT]->(n);

LOAD CSV WITH HEADERS FROM 'file:///points.csv' AS row

WITH row WHERE NOT row.points IS null

MERGE (n:Points {score: toInteger(row.points)})

WITH row, n

MATCH(w:Wine {ID: row.ID})

MERGE (w)-[:EARNED]->(n);

LOAD CSV WITH HEADERS FROM 'file:///price.csv' AS row

WITH row WHERE NOT row.price IS null

MERGE (n:Price {costbucket:row.pricebucket, maxvalue:toInteger(row.maxvalue)})

WITH row, n

MATCH(w:Wine {ID: row.ID})

MERGE (w)-[:COSTS\_BETWEEN]->(n);

LOAD CSV WITH HEADERS FROM 'file:///region1.csv' AS row

WITH row WHERE NOT row.region\_1 IS null

MERGE (n:Region1 {name: row.region\_1})

WITH row, n

MATCH(w:Wine {ID: row.ID})

MERGE (w)-[:FROM]->(n);

LOAD CSV WITH HEADERS FROM 'file:///province.csv' AS row

WITH row WHERE NOT row.province IS null

MERGE (n:Province {name: row.province})

WITH row, n

MATCH(w:Wine {ID: row.ID})

MATCH(r:Region1 {name: row.region\_1})

MERGE (w)-[:FROM]->(r)-[:IN]->(n);

LOAD CSV WITH HEADERS FROM 'file:///country.csv' AS row

WITH row WHERE NOT row.country IS null

MERGE (c:Country {name: row.country})

WITH row, c

MATCH(w:Wine {ID: row.ID})

MATCH(r:Region1 {name: row.region\_1})

MATCH(p:Province {name: row.province})

MERGE (w)-[:FROM]->(r)-[:IN]->(p)-[:IN]->(c);