```
def negate(literal):
  return literal[1:] if literal.startswith("-") else "-" + literal


def pl_resolution(kb, query):
  # Print Knowledge Base
  print("Knowledge Base:")
  print("[")
  for clause in kb:
      print(f"  {clause},")
  print("]")

  # Print Query
  print("Query:")
  print(query)

  # Append the negated query to the KB
  clauses = kb[:]
  clauses.append([negate(q) for q in query])


  new = set()

  while True:
      n = len(clauses)
      pairs = [(clauses[i], clauses[j]) for i in range(n) for j in range(i + 1, n)]

      for (ci, cj) in pairs:
          resolvents = pl_resolve(ci, cj)
          if [] in resolvents:
              return True
          for res in resolvents:
              new.add(tuple(sorted(res)))
```

```
        new_clauses = [list(c) for c in new if list(c) not in clauses]
        if not new_clauses:
            return False

        clauses.extend(new_clauses)


def pl_resolve(ci, cj):
    resolvents = []
    for di in ci:
        for dj in cj:
            if di == negate(dj):
                new_clause = list(set(ci + cj))
                new_clause.remove(di)
                new_clause.remove(dj)
                resolvents.append(new_clause)
    return resolvents


# ---------------------------------------------
# Input from your screenshot
# ---------------------------------------------
knowledge_base = [["~P", "Q"], ["P"], ["~Q", "R"], ["~R"]]
query = ["R"]


# ---------------------------------------------
# Execution with Output Display
# ---------------------------------------------
result = pl_resolution(knowledge_base, query)
```

```
print(f"Knowledge Base: {knowledge_base}")
print(f"Query: {query}")


if result:
  print("The query is satisfiable.")
else:
  print("The query is not satisfiable.")
```