Project 2
Diego, Rio

Query 1 Design:

For each drug, compute the number of genes and the number of diseases associated with the drug. Output results with the top 5 number of genes in a descending order.

Overall design:
Compound: gene/disease pairs that reduce to compound: # associations

Walkthrough design:
We approach it as a Numerical summarization problem. Finding out how to break down the total list of relationships into top 5 highest gene counts.

Input: Edges.tsv

Node	Relationship	Node

We want to first focus on isolating the edges such that we are focusing on the compounds. We can filter by finding these relationships.
Compound to Gene.
Compound to Disease.
We then create two clusters unique to each relationship.

Compound A	C G	Gene
Compound A	C D	Disease

When we find these relationships we ignore the contents of each disease or gene, rather we convert those into counters. The counter represents the appearance of the gene. We **map** each relationship to convert it into Compound Name: 1.

Compound A	1 (relation to a gene)
Compound A	1 (relation to a disease)

Then we can **reduce** all the compound key values into one Compound key with accumulated value.

Compound A	N (relations to genes)
Compound A	M (relation to diseases)

Finally we **join** the two compound lists so that we can have a total number of appearances of every compound. The two total relationships counts will be turned into a pair.

Compound A	(N,M)

We can sort this new container with the highest value N, then return the first 5 of the sorted (descending) container.

Query 1 Pseudocode:

```
Start SparkContext
Input(RDD a <- {{node1, rel1, nodeM}...{nodeN, relM, nodeK}})
        B <- Filter a s.t it only provides Compound to Gene Relationships
        //Mappings
        B <- Map each 3-tuple {Compound, Rel, Gene} to 2 tuple {Compound, Gene}for
        simplicity and to skip over repeats
        B <- Map every pair {Compound, Gene} ->{Compound, 1}
        //Reducing
        B <- Reduce all appearances of compounds to total count value
                {Compound 1, 1}...{Compound 1, 1} -> {Compound 1, n}
        //repeat for Compound to Disease relationships
        C <- Filter a s.t it only provides Compound to Disease Relationships
        //Mappings
        C <- Map each 3-tuple {Compound, Rel, Disease} to 2 tuple {Compound, Disease}for
        simplicity and to skip over repeats
        C <- Map every pair {Compound, Gene} ->{Compound, 1}
        //Reducing
        C <- Reduce all appearances of compounds to total count value
                {Compound 1, 1}...{Compound 1, 1} -> {Compound 1, n}

        D <- Join B and C {Compound: n Genes, m Diseases}
        D <- Sort B based on highest Gene relationship count
        Return first 5 elements of D
        Print Results
        Stop SparkContext
```

Query 2 Design:

Compute the number of diseases associated with 1, 2, 3, …, n drugs. Output results with the top 5 number of diseases in a descending order.

Overall Design:
Compound: Disease pairs that reduce to #diseases: 1 compound and then
To #compound:#diseases

Walkthrough Design:
This Query is also a numerical summarization problem. With the goal of relating number of drugs with number of associated diseases

Input: Edges.tsv

Node	Relationship	Node

We want to first focus on isolating the edges such that we are focusing on the compounds. We can filter by finding these relationships.
Compound treats Disease
Compound palliates Disease

We then create one cluster containing both: C _ D

Compound A	C D	Disease A

We can now minimize so that we don't operate on the types of disease:
Map 3 tuple to pair and Inverse key Value to value:key; this is to reduce an inversion later

Disease A	Compound A

Since a compound may be able to treat and palliate a disease
It can repeat
We can filter out for repeats to avoid overcounting
Then Mapping to the compound to a count of 1

Disease A	1 (associated Compound)

Then Reduce all Disease A pairs to be 1 accumulated pair

Disease A	N (total associated Compounds)

Now We can invert the pairing again so that we can focus on which how many compounds map to 1 disease

N (total associated Compounds)	Disease A

In a similar way to how we Accumulated count N we can count how many compounds map to how many disease

N	1 (Disease)

Reducing N compounds per 1 disease to N compounds to M diseases

N (Compounds)	M(Diseases)

Finally, We can sort all pairings so that we can find the compounds with the most diseases associated with. We return the top 5 count of compounds with highest disease associations

Query 2 Pseudocode:

Start SparkContext
Input(RDD A ← {{node1, rel1, nodeM}...{nodeN, relM, nodeK}})
    B ← Filter A s.t it provides Compound to all Disease Relationships

    //mapping
    B ← Map 3 tuple of to 2 tuple: {node, rel, node} -> {Compound, Disease}
    B ← Filter out repeats
    B ← Map 2 tuple to inverse 2 tuple: {Compound, Disease} ->{Disease, Compound}
    B ← Map {Disease,Compound} ->{Disease, 1}

    //reduce to get Diseases with n Drug Relations
    B ← Reduce All n {Disease, 1 } appearances to {Disease, n }

    //Mapping again
    B← inverse map {Disease, n } ->1 {n, Disease}
    B← Map {n, Disease} -> {n, 1}

    //Reduce again
    B ← Reduce all m {n,1} -> 1 {n, m }

    B ← Sort by highest disease count

Return first 5 B elements1
Print Results
Stop SparkContext

Query 3 Design:

Get the name of drugs that have the top 5 number of genes. Output the results.

Overall Design:
Compound:Gene Pairs to Gene:Compound pairs and then Gene:List(Compounds)
To Gene:# compounds, then using inverted pairs #compounds:Gene to sort

Walkthrough Design
Another numerical summarization problem with the addition of mapping the compound ID to its name.

Input: Edges.tsv

Node	Relationship	Node

We want to first focus on isolating the edges such that we are focusing on the compounds. We can filter by finding these relationships.
Compound u Gene.
Compound d Gene
Compound b Gene

We then create one cluster that holds all these relationships.

Compound A_id	C _ G	Gene

Input: Nodes.tsv
Format:

Node ID	Node Name	Node Type

For the purpose of obtaining the names of the nodes we are using,
We focus on filtering using the Nodes.tsv File:
1 - Only using those with type Compounds to get
Compound ID to Compound Name
2 - Only using those with type genes to get

Gene ID to Gene Name
Now we have 2 containers with just filtered types and mapped to Node ID-Name Mappings

Compound ID	Compound Name
Gene ID	Gene Name

Now we go back to our relationship container
We map our relationships so that we skip over relationship type and invert the relationship
Also filter out duplicates

Gene ID	Compound ID

Then We call group by key into lists:

Gene ID	{Compound ID_A, Compound ID_B, …}

Then Map list to its set size

Gene ID	N (size of elements in list)

We can invert, the pair and then we group the values into a list

N compounds	{Gene ID_A, …Gene ID_Z}

Now we can sort our container by highest N and find the top 5 Genes

Query 3 PseudoCode

Start SparkContext

Input(RDD Edges ← (Node ID, Relationship, Node ID), RDD Nodes ← (Node ID, Node Name))

A← filter Edges st we only get compound ot gene edgesRDD of (Compound ID, _, Gene ID)

B← Filter Nodes s.t we get Gene Nodes RDD of (Gene ID: Gene Name)

A← Map A s.t we skip ignore _ relationship type, and make RDD of inverted pairs of (Gene, Compound), then filter out duplicates

A← Map A s.t we count unique compounds per gene: RDD of (Gene, # compounds)

A← Invert Pairs then group by number of compounds, RDD: (#compounds: List(Genes))

A← Sort Pairs by key in descending order

Return top 5 key-value pairs
Print Results
Stop SparkContext

Query Results

Query 1 Results
Compound: Compound::DB08865
	- 580 genes and 1 diseases
Compound: Compound::DB01254
	- 558 genes and 1 diseases
Compound: Compound::DB00997
	- 528 genes and 17 diseases
Compound: Compound::DB00390
	- 521 genes and 2 diseases
Compound: Compound::DB00170
	- 521 genes and 0 diseases

Query 2 Results
Disease::DOID:10763 associated with 70 compounds
  - disease name: hypertension
Disease::DOID:2531 associated with 53 compounds
  - disease name: hematologic cancer
Disease::DOID:1612 associated with 43 compounds
  - disease name: breast cancer
Disease::DOID:2841 associated with 40 compounds
  - disease name: asthma
Disease::DOID:3393 associated with 38 compounds
  - disease name: coronary artery disease

Query 3 Results
CYP3A4 associated with 516 compounds
    Compounds: {refer to terminal output}
CYP2D6 associated with 293 compounds
    Compounds: {refer to terminal output}
ABCB1 associated with 272 compounds
    Compounds: {refer to terminal output}
CYP2C9 associated with 262 compounds
    Compounds: {refer to terminal output}
CYP1A2 associated with 230 compounds
    Compounds: {refer to terminal output}