Proyecto Rune Factory 4

```
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX ex:  <http://ex.org/a#>
```

1) **Items that have the word “cabbage” in the name**

```
SELECT * WHERE {
  ?uri a ex:Item .
  ?uri rdfs:label ?name .
  FILTER ( regex(?name, "Cabbage") )
}
```

2) **Map Locations that have “Wooly” monsters.**

```
SELECT ?area ?map ?floor (GROUP_CONCAT(?subarea; SEPARATOR=", ") AS ?section)
WHERE {
  ?loc a ex:MapLocation ;
        ex:has ?wooly ;
              ex:mainArea ?area ;
              ex:mapURL ?map ;
       ex:floor ?floor ;
              ex:subArea ?subarea .
  ?wooly a ex:Monster ;
       rdfs:label 'Wooly' .
}
       GROUP BY ?area ?map ?floor
```

3) **Map locations that have monsters, and the name of a monster found there as example.**

```
SELECT ?location (SAMPLE(?name) AS ?monsterName) WHERE {
  ?loc a ex:MapLocation ;
       ex:mainArea ?location .

  ?loc ex:has ?a .
  ?a a ex:Monster .
  ?a rdfs:label ?name .
}
GROUP BY ?location
```

4) **Monsters that like the item “Wine” as a gift**

```
SELECT ?npc ?name WHERE {
  ?npc a ex:Monster .
  ?npc rdfs:label ?name .
```

```
  ?npc ex:like1 ?l1 .
  OPTIONAL{?npc ex:like2 ?l2}
  OPTIONAL{?npc ex:like3 ?l3}
  OPTIONAL{?npc ex:like4 ?l4}
  FILTER(?l1 = 'Wine' || ?l2 = 'Wine' || ?l3 = 'Wine' || ?l4 = 'Wine')
}
```

5) **Return the name and liked gifts of the monsters that drop “Honey” in battle.**

```
SELECT ?name ?likes WHERE {
  ?uri a ex:NPC .
  ?uri rdfs:label ?name .
  {?uri ex:drop1 ?item .}
  UNION {?uri ex:drop2 ?item .}
  UNION {?uri ex:drop3 ?item .}
  UNION {?uri ex:drop4 ?item .}
  ?item rdfs:label "Honey" .
  ?uriM a ex:Monster .
  ?uriM owl:sameAs ?uri .
  {?uriM ex:like1 ?likes .}
  UNION {?uriM ex:like2 ?likes .}
  UNION {?uriM ex:like3 ?likes .}
  UNION {?uriM ex:like4 ?likes .}
}
```

6) **Return the name of the monsters that produce “Egg (S)” as friendly NPCs and where to find them (world area and sub area).**

```
SELECT ?name ?mainArea ?subArea WHERE {
  ?place a ex:MapLocation;
  ex:mainArea ?mainArea;
  ex:has ?monster.
  OPTIONAL {?place ex:subArea ?subArea.}
  ?monster a ex:Monster;
  rdfs:label ?name;
  owl:sameAs ?npc.
  ?npc ex:produce ?item.
  ?item rdfs:label "Egg (S)" .
}
```

7) **Return all the cooked dishes that are liked by monsters and made out of fish.**

```
SELECT DISTINCT ?monster ?itemName ?mName1 WHERE {
  ?monster a ex:Monster .
  {?monster ex:like1 ?l .}
  UNION {?monster ex:like2 ?l .}
  UNION {?monster ex:like3 ?l .}
```

```
  UNION {?monster ex:like4 ?l .}
  ?item a ex:Item .
  ?item rdfs:label ?itemName .
  ?item ex:obtainMethod "Cooked" .
  ?recipe a ex:Recipe .
  ?recipe ex:name ?name .
  ?recipe ex:material1 ?m1 .
  ?m1 ex:obtainMethod "Fished" .
  ?fish a ex:Fish .
  ?fish ex:name ?fishName .
  FILTER(?name = ?itemName &&(?l = ?itemName ))
}
```

8) **Return the name and the main obtained methods for the items liked by the "Sarcophagus" creature.**

```
SELECT DISTINCT ?l ?o WHERE {
  ?monster a ex:Monster .
  ?monster rdfs:label "Sarcophagus" .
  {?monster ex:like1 ?l .}
  UNION {?monster ex:like2 ?l .}
  UNION {?monster ex:like3 ?l .}
  UNION {?monster ex:like4 ?l .}
  ?item1 a ex:Item .
  ?item1 rdfs:label ?l .
  ?item1 ex:obtainMethod ?o .
}
```

9) **Return the names of all items which can be dropped by monsters and also obtained by mining.**

```
SELECT ?drop WHERE {
  ?npc a ex:NPC .
  {?npc ex:drop1 ?l .}
  UNION {?npc ex:drop2 ?l .}
  UNION {?npc ex:drop3 ?l .}
  UNION {?npc ex:drop4 ?l .}
  ?l rdfs:label ?drop .
  ?item1 a ex:Item .
  ?item1 rdfs:label ?drop .
  ?item1 ex:obtainMethod 'Mined' .
}
```

10) **Find items liked by monsters that are obtained by the "Crafted" method, and return their names and recipes.**

```
SELECT DISTINCT (?l AS ?itemName) ?material1 ?material2 ?material3 ?material4
WHERE {
  ?monster a ex:Monster .
  {?monster ex:like1 ?l .}
  UNION {?monster ex:like2 ?l .}
  UNION {?monster ex:like3 ?l .}
  UNION {?monster ex:like4 ?l .}
  ?item a ex:Item .
  ?item rdfs:label ?l .
  ?item ex:obtainMethod 'Crafted' .
  ?recipe a ex:Recipe .
  ?recipe ex:name ?l .
  ?recipe ex:material1 ?material1 .
  OPTIONAL {?recipe ex:material2 ?material2 .}
  OPTIONAL {?recipe ex:material3 ?material3 .}
  OPTIONAL {?recipe ex:material4 ?material4 .}
}
```

11) Find fish that appear in only one season, and the amount of cooking recipes that use them as ingredients.

```
SELECT ?fishName (COUNT(?recipe) AS ?recipeCount) WHERE {
  # Get season-exclusive fish:
  ?fish a ex:Fish ;
       ex:name ?fishName ;
       ex:spring ?SPRrate ;
             ex:summer ?SUMrate ;
             ex:autumn ?AUTrate ;
             ex:winter ?WINrate .
  BIND ( ?SPRrate + ?SUMrate + ?AUTrate + ?WINrate AS ?total)
  FILTER (?total!=0 && ?total IN (?SPRrate, ?SUMrate, ?AUTrate, ?WINrate))
  ?fishURI a ex:Item ;
      rdfs:label ?fishName .

  # Get recipes with those as ingredients:
  ?recipe a ex:Recipe .
  {?recipe ex:material1 ?item .}
  UNION {?recipe ex:material2 ?item .}
  UNION {?recipe ex:material3 ?item .}
  UNION {?recipe ex:material4 ?item .}
  UNION {?recipe ex:material5 ?item .}
  UNION {?recipe ex:material6 ?item .}
  FILTER (?item = ?fishURI)
}
```

```
GROUP BY ?fishName
```

12) Return the name and obtain method of every ingredient for “Marionetta”’s favorite gifts.

```
SELECT (?gift AS ?giftRecipe) ?ingredient ?method ?details WHERE {
  # Search for Marionetta's favorites
  ?marionetta a ex:Monster;
              rdfs:label "Marionetta" .
  {?marionetta ex:like1 ?gift .}
  UNION {?marionetta ex:like2 ?gift .}
  UNION {?marionetta ex:like3 ?gift .}
  UNION {?marionetta ex:like4 ?gift .}

  # Get ingredients:
  ?recipe a ex:Recipe ;
            ex:name ?gift .
  {?recipe ex:material1 ?item .}
  UNION {?recipe ex:material2 ?item .}
  UNION {?recipe ex:material3 ?item .}
  UNION {?recipe ex:material4 ?item .}
  UNION {?recipe ex:material5 ?item .}
  UNION {?recipe ex:material6 ?item .}

  # Get method:
  ?item ex:obtainMethod ?method .
  ?item rdfs:label ?ingredient .

  # Obtaining details for each case of methods:
  # (if farmed, show seeds that should be planted)
  OPTIONAL {?crop a ex:Crop;rdfs:label ?ingredient;ex:pickupSeed ?seed .}
  # (if purchased, get buying price)
  OPTIONAL {?item ex:buy ?price .}
  # (if cooked, get furniture needed to make recipe)
  OPTIONAL {?itemRecipe a ex:Recipe ;
                        ex:itemURI ?item ;
                        ex:type ?utensil .
      OPTIONAL {?utensil rdfs:label ?category.}
      BIND (COALESCE (?category, "Chemistry Set") AS ?furniture )}
  # (if produced, return all monster options that give that item)
  OPTIONAL {
      SELECT ?item (GROUP_CONCAT(?monsterName; SEPARATOR=", ") AS ?monsters)
{
      ?monster a ex:NPC ; ex:produce ?item ; rdfs:label ?monsterName .}
```

```
      GROUP BY ?item
  }

  # Create a single column with the corresponding details:
  BIND ( COALESCE(
      IF(?method = "Farmed", concat("Plant ", ?seed), 1/0),
      IF(?method = "Purchased", concat(str(?price), "G in General Store"),
1/0),
      IF(?method = "Cooked" , concat("Use ", ?furniture), 1/0),
      IF(?method = "Produced", concat("Tame ", ?monsters), 1/0),
      # can be extended to consider picked up, dropped, fished, etc.
      1/0
      ) AS ?details)
}
```

13) Return the name and where/how to obtain the ingredients required to make "Royal Curry".

```
SELECT ?ingredient ?method ?details WHERE {
  # Get ingredients:
  ?recipe a ex:Recipe ;
            ex:name "Royal Curry" .
  {?recipe ex:material1 ?item .}
  UNION {?recipe ex:material2 ?item .}
  UNION {?recipe ex:material3 ?item .}
  UNION {?recipe ex:material4 ?item .}
  UNION {?recipe ex:material5 ?item .}
  UNION {?recipe ex:material6 ?item .}

  # Get method:
  ?item ex:obtainMethod ?method .
  ?item rdfs:label ?ingredient .

  # Obtaining details for each case of methods:
  # (if farmed, show seeds that should be planted)
  OPTIONAL {?crop a ex:Crop;rdfs:label ?ingredient;ex:pickupSeed ?seed .}
  # (if purchased, get buying price)
  OPTIONAL {?item ex:buy ?price .}
  # (if cooked, get furniture needed to make recipe)
  OPTIONAL {?itemRecipe a ex:Recipe ;
                              ex:itemURI ?item ;
                        ex:type ?utensil .
      OPTIONAL {?utensil rdfs:label ?category.}
      BIND (COALESCE (?category, "Chemistry Set") AS ?furniture )}
  # (if produced, return all monster options that give that item)
```

```
  OPTIONAL {
      SELECT ?item (GROUP_CONCAT(?monsterName; SEPARATOR=", ") AS ?monsters)
{
      ?monster a ex:NPC ; ex:produce ?item ; rdfs:label ?monsterName .}
      GROUP BY ?item
  }

  # Create a single column with the corresponding details:
  BIND ( COALESCE(
      IF(?method = "Farmed", concat("Plant ", ?seed), 1/0),
      IF(?method = "Purchased", concat(str(?price), "G in General Store"),
1/0),
      IF(?method = "Cooked" , concat("Use ", ?furniture), 1/0),
      IF(?method = "Produced", concat("Tame ", ?monsters), 1/0),
      1/0
      ) AS ?details)
}
```

14) The name and price of the item with the highest selling price that can be made using a "Magic Crystal".

```
SELECT ?recipeName ?price WHERE {
  ?recipe a ex:Recipe ; ex:name ?recipeName .
  {?recipe ex:material1 ?item .}
  UNION {?recipe ex:material2 ?item .}
  UNION {?recipe ex:material3 ?item .}
  UNION {?recipe ex:material4 ?item .}
  UNION {?recipe ex:material5 ?item .}
  # Recipes that need Magic Crystal:
  {?item rdfs:label "Magic Crystal" .}
  # Recipes that accept any item of Magic Crystal's type:
  UNION {
       ?mcrystal a ex:Item ;
       rdfs:label "Magic Crystal" ;
       rdfs:subClassOf ?crystalID .
       ?crystalType a ex:ItemType ;
       owl:sameAs ?crystalID ;
       rdfs:label ?typeName .
      ?item rdfs:subClassOf ?crystalID .
  }
  ?item ex:sell ?price .

}
ORDER BY DESC(?price)
```

15) **The name and items needed to forge weapons that cannot be bought (have buying price 0).**

```
SELECT ?weaponRecipe ?material1Name ?material2Name ?material3Name
?material4Name ?material5Name WHERE {
  ?recipe a ex:Recipe .
  ?recipe ex:type ?category .
  ?category ex:info "Weapons" .
  ?recipe ex:itemURI ?item .
  ?item ex:buy 0 .
  ?recipe ex:name ?weaponRecipe .
  ?recipe ex:material1 ?m1 .
  ?m1 rdfs:label ?material1Name .
  OPTIONAL{
      ?recipe ex:material2 ?m2 .
      ?m2 rdfs:label ?material2Name .
      ?recipe ex:material3 ?m3 .
      ?m3 rdfs:label ?material3Name .
      ?recipe ex:material4 ?m4 .
      ?m4 rdfs:label ?material4Name .
      ?recipe ex:material5 ?m5 .
      ?m5 rdfs:label ?material5Name .
  }
}
```

16) **The best accessory that you can make (given that you are level 40) that gives you an attack bonus when equipped.**

```
SELECT ?recipe ?recipeLvl WHERE {
  ?category a ex:ItemCategory .
  ?category rdfs:label "Accessory" .
  ?recipe a ex:Recipe .
  ?recipe ex:type ?category .
  ?recipe ex:itemURI ?item .
  ?recipe ex:name ?recipeName.
  ?item ex:bonusType "Atk" .
  ?recipe ex:level ?recipeLvl .
  FILTER( ?recipeLvl <= 40 )
}
ORDER BY DESC(?recipeLvl)
LIMIT 1
```

17) **The item with the highest selling price that can be found in the first floor of “Idra Cave”, considering different obtaining methods.**

```
SELECT ?itemName ?method ?price WHERE {
  ?place a ex:MapLocation .
  ?place ex:mainArea "Idra Cave" .
  ?place ex:floor "1F" .

  # Items dropped by monsters
  ?place ex:has ?monster .
  ?monster a ex:Monster .
  {?monster ex:drop1 ?item .}
  UNION {?monster ex:drop2 ?item .}
  UNION {?monster ex:drop3 ?item .}
  UNION {?monster ex:drop4 ?item .}

  # Items found by picking up
  UNION {
      ?item a ex:Item .
      ?place ex:has ?item .
  }
  # Items found by hitting ores
  UNION {
      ?geode a ex:Geode .
      ?place ex:has ?geode .
      {?geode ex:drop1 ?item .}
      UNION {?geode ex:drop2 ?item .}
      UNION {?geode ex:drop3 ?item .}
      UNION {?geode ex:drop4 ?item .}
  }
  # Find the best selling price and method
  ?item ex:sell ?price .
  ?item ex:obtainMethod ?method .
  ?item rdfs:label ?itemName .
}
ORDER BY DESC(?price)
LIMIT 1
```