title: Create a condition function for the rule weight: 20

Create a file `condition.js` for the condition part of the rule.

Complete these steps:

1. Create the`condition.js` file and include the object, relation, and dotenv modules.

```
require('dotenv').config({path: __dirname + '/.env'});
var KnowledgeObject = require('./sdk/object');
var KnowledgeRelation = require('./sdk/relation');
```

Tip: `__dirname` is required for Cloud Functions to find the file in the container. The condition part of the rule finds the owner of a house when the agent is notified that the front door has opened. Before sending an alert to the home owner, the agent checks that the house is unoccupied. 2. Add a function to `condition.js` that finds the owner of a house from a door ID.

```
function getHouseAndPersonForDoor(doorId) {
  console.log('in getHouseAndPersonForDoor');
  var door, house, owner;
  return KnowledgeObject.retrieve(doorId).then((doorObj) => {
    door = doorObj;
    console.log('Door id', door.id);
    // Get the house of the door
    return door.both('has-as-part');
  }).then((parts) => {
    house = parts[0];
    console.log('House', house.id);
    // Get the owner of the house
    return house.both('ownership');
  }).then((owners) => {
    owner = owners[0];
    console.log('Owner', owner.id);
    return new Promise((res, rej) => {
      res([door, house, owner]);
    });
  }).catch((err) => {
    console.log('Error: ' + err);
  });
}
```

In this function, which is given a specific door ID, the function traverses the `has-as-part` relationship to the house object. The function traverses the `ownership` relationships to the owner. The function returns the person, the house and the front door.

2. Create a NodeJS function to check that the update event referred to a door. The agent is not interested in updates to houses or owners.

```
function checkType(event, type) {
  var eventType = event[0]['type'];
  if (eventType == type) {
    return true;
  } else {
    return false;
  }
}
```

3. Create the main function that checks if the owner is away when the door is opened.

```
function main(event, callback) {
  console.log('in condition main');
  var doorId = event[0]['id'];
  console.log('got door id as ' + doorId);
  if (checkType(event, 'Door')) {
    return getHouseAndPersonForDoor(doorId).then((objects) => {
      var door = objects[0];
      var house = objects[1];
      var owner = objects[2];
      // if door is open and owner isn't at home
      if (door.attributes.isOpen &&
        (owner.attributes['longitude'] != house.attributes['longitude'] ||
        owner.attributes['latitude'] != house.attributes['latitude'])) {
        console.log("door is open and owner isn't home - return True");
        callback(true);
      } else {
        console.log("door is closed or owner is at home - return False");
        callback(false);
      }
    });
  } else {
    console.log("update wasn't on a door - return False");
    callback(false);
  }
}
```

The function returns `True` or `False`.
4. Add code to allow you test the condition rule locally as well as on IBM Cloud Functions.

```
// To support testing locally and running in Cloud Functions
if (require.main === module) {
  console.log("running locally")
  // parse the input from the command line $ node index.js 123
  doorID = process.argv[2]
  console.log(process.argv)
  main({ results: [{ id: doorID, type: 'Door' }] })
    .then((result) => {
      console.log("action is done running success");
      console.log(JSON.stringify(result));
    })
    .catch((err) => {
      console.log("action is done running error");
      console.log(JSON.stringify(err));
    });
} else {
  console.log("running in openwhisk")
  exports.main = main;
  exports.checkType = checkType;
}
```

5. Save your changes to `conditon.js`.

> **What to do next?**
> Create the action part of the rule.