Additional cypher examples:

Consider the following graph, which represents a social network for recommendations on businesses. It has the following nodes (each with a single label):

<table>
<thead>
<tr>
<th>USER</th>
<th>RESTAURANT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nickname</td>
<td>Name</td>
</tr>
<tr>
<td>Name</td>
<td>Address</td>
</tr>
<tr>
<td>Hobbies</td>
<td>Telephone</td>
</tr>
<tr>
<td>Type</td>
<td></td>
</tr>
</tbody>
</table>

The graph has the following edges:

FRIEND_OF: A directional single label link between nodes with a label :USER. It has a single property call "Trust".

REVIEW: A directional single label link between nodes with label :USER to nodes with label :RESTAURANT. It has two properties called "Ranking" and "Text".

Write the following queries:

1) Find all users whose FRIEND_OF distance from a user called "Moshe" is up to 5 hops and have recommended a restaurant called "Greg".

   MATCH (:RESTAURANT {Name:"Greg"})<-[r:REVIEW]-{u:USER}-[:FRIEND_OF*..5]-(m:USER {Name:"Moshe"})
   RETURN u.Nickname

2) Find the average rating of a restaurant called "Greg" provided by all users whose FRIEND_OF distance from a user called "Moshe" is up to 5 hops and have recommended this restaurant.

   MATCH (:RESTAURANT {Name:"Greg"})<-[:REVIEW]-{u:USER}-[:FRIEND_OF*..5]-(m:USER {Name:"Moshe"})
   RETURN avg(r.Ranking)

3) How many users are friends of "Moshe"?

   MATCH (f:USER)-[:FRIEND_OF]->(:USER {Name:"Moshe"})
   RETURN count(f)

4) How many friends on average each user has.

   MATCH (u:USER)
   WITH count(u) as un
   MATCH ()-[f:FRIEND_OF]->()
   RETURN 1.0*count(f)/un