

Übung 3

Das Schema entsprach bereits den geforderten Bedingungen.
Nicht mehr benötigte Einträge wurden entfernt.

Zum integrieren der GO wurde die Tabelle "go" um den Eintrag "namespace" erweitert und eine weitere Tabelle "go_is_a" eingeführt. Die Einträge wurden mit einem Python-Skript vorbereitet und dann Eingefügt/Aktualisiert.

Um Query 6 schneller zu beantworten verwenden wir ein Tabelle "dist_nodes" welche für jeden GO-Term alle Terme enthält, die (direkt oder indirekt) durch "is_a" verbunden sind.

Queries

(1)

```
SELECT count(id) FROM genes;  
⇒ 59852
```

(2)

```
SELECT count(DISTINCT (start_position, end_position)) AS positions  
FROM refseqs  
GROUP BY gene_id  
ORDER by positions DESC  
LIMIT 1;  
⇒ 37
```

(3)

```
WITH singles AS (  
    SELECT count(DISTINCT (start_position, end_position))  
    FROM refseqs  
    GROUP BY gene_id  
    HAVING count(DISTINCT (start_position, end_position)) = 1  
)  
SELECT count(*) FROM singles;  
⇒ 4061
```

(4)

```
SELECT count(id) FROM go WHERE namespace IS NOT NULL;
```

⇒ 42832

```
SELECT namespace, count(id)
  FROM go
 WHERE namespace IS NOT NULL
 GROUP BY namespace;
```

namespace	count
biological_process	28688
molecular_function	10238
cellular_component	3906

(5)

```
WITH temp AS (
  SELECT genes.id AS g_id, count(go_id) AS assigned
    FROM genes LEFT JOIN genes_go ON genes.id = genes_go.gene_id
   GROUP BY genes.id
)
SELECT assigned, count(g_id)
  FROM temp
 GROUP BY assigned
 ORDER BY assigned;
```

(6)

Als PL/pgSQL-Funktion "fill_assignment" umgesetzt.

Wenn Tabelle "assignment" leer ist (TRUNCATE assignment;) neu füllen mit
SELECT * FROM fill_assignment();

Laufzeit war ca. 22 s.

Für GO:0005150:

```
SELECT genes FROM assignment WHERE go_id = 'GO:0008150';
⇒ 16707
```

Falls die vormaterialisierten Daten in "dist_nodes" nicht zulässig sind kann auch alles neu berechnet werden (Laufzeit ca. 140 s):

```
TRUNCATE assignment;
TRUNCATE dist_nodes;
SELECT * FROM fill_nodes_assignment();
```