《EQL--特别简单易学的知识图谱查询语言，实现高速精准搜索》

刘韩 刘山涛
北京柠檬花科技有限公司

2020.3.15

论文摘要：

EQL，又称极简查询语言(Extremely Simple Query Language)，可以普遍用于知识图谱、精准搜索、强人工智能、数据库、智能音箱、专利检索等领域。EQL在设计上采用极简原则，追求简单易学，使所有人都可以快速掌握。

在底层实现上，通过采用人工智能中的NLP和知识图谱等技术，以及大数据和搜索引擎领域的技术，确保实现高速精准的信息搜索，从而帮助用户更好地挖掘出数据资产中蕴含的财富。

EQL语言与λ演算的互相转化，揭示了EQL语言的数学本质，为EQL语言的严密性和逻辑完整性奠定了坚实的基础。

从本质上来说，任何一个陈述句都等价于一个spo事实语句，spo事实语句的数学抽象形式是s:p:o(q1:v1,q2:v2.....qn:vn)，s:p:o为主语、谓语、宾语，或实体、属性、属性值，(q1:v1,q2:v2.....qn:vn)称为修饰语，修饰语补充s:p:o部分更多的细节，n为0或自然数；任何一个疑问句，等价于一个或多个spo疑问语句，每个spo疑问语句的基本形式是将spo事实语句中的任一项替换为?x形式，例如：s:p:?x(q1:v1,q2:v2),或s:p:o(q1:?x,q2:v2,q3:v3)。

以上命题，可以简称为刘韩定理（Alan Liu Theorem),是EQL语言的理论基础。谨以刘韩定理，致敬伟大的哥德尔不完备性定理(Godel Incompleteness Theorem)，和丘奇-图灵定理（Church-Turing Theorem)。

数百亿条spo事实语句，可以构成世界常识知识图谱。EQL的本质，就是对spo疑问语句加以扩展，让人们极其方便地访问知识图谱中的信息。

EQL语言和完善的世界常识知识图谱系统，可以共同构成未来强人工智能的基础，弥补目前人工智能系统对世界常识缺少理解引发的缺憾。EQL语言不仅可以供人类使用，也可以作为机器人之间的数据查询和数据交换的基本语言。

1.引言

人类社会进入信息时代以来，人类拥有的数据呈指数级别增长。特别是互联网和移动互联网的发展，更加速了数据爆炸的趋势。然而，人们查询数据的语言已经有多年没有进步，最常用的数据查询语言仍是1974年由Boyce和Chamberlin提出的SQL语言，SQL语言算是专业的关系数据库查询语言，但是除了程序员和数据库管理员外，基本上很少用户能够掌握。

近年来，在Google公司的引领之下，知识图谱领域有了较大进展，知识图谱中的知识和数据在数量和质量上都有了很大提升。但是，知识图谱的查询语

言仍然是用户使用中的重要瓶颈，知识图谱领域常用的SPARQL语言于2008年首次面世，复杂性与SQL语言基本相当。用于图数据库的Cypher查询语言，于2011年面世，和SPARQL难度差不多，对普通人来说，仍然很难掌握。

本文论述的EQL，极简查询语言(Extremely Simple Query Language)，可以普遍用于知识图谱、精准搜索、强人工智能、数据库、智能音箱、专利检索等领域。在查询语言设计上采用极简原则，追求简单易学，使所有人都可以快速掌握。在底层实现上，通过采用人工智能中的NLP自然语言理解和知识图谱等技术，以及大数据和搜索领域的技术，EQL确保实现高速精准的信息搜索，从而帮助用户更好地挖掘出数据资产中蕴含的财富。

为了方便非专业读者，本文力求将简单的内容放在前面的章节，将难度大的内容放在文章后部。

为致敬SQL发音为sequel的传统，EQL发音为equal，国际音标['i:kwəl]。EQL象征着人类对自由平等的追求和向往，帮助普通人在信息时代得到平等的数据赋能。

2.EQL的基本语法

让我们从一些简单的例子开始了解EQL。每个案例，我们会先给出自然语言描述的查询问题，在"EQL:"之后c一句EQL语句，在“ANS:"之后是返回的结果：

案例1：理解spo语句

问题:谁获得了2016年的诺贝尔文学奖？
EQL：?:所获奖项:诺贝尔文学奖（日期:2016年）
ANS：鲍勃迪伦

在EQL配套的知识图谱系统中，一条事实数据是以如下抽象形式存放的：

s:p:o(q1:v1,q2:v2,q3:v3,.....，qn:vn)

这样的抽象形式称为一行spo语句。

全部的spo语句即构成我们查询的知识图谱系统，完善的知识图谱系统可以包含几百亿条spo语句，存储人类绝大多数学科的基本常识。EQL语言和完善的世界常识知识图谱系统可以共同构成未来强人工智能的基础，弥补目前人工智能系统对世界常识缺少理解的缺憾。EQL语言不仅可以供人类使用，也可以作为机器人之间的数据查询和数据交换的基本语言。

EQL语言对应的数据底层存储系统，可以有多种选择，如SQL数据库、RDF数据库、图数据库、MongoDB等NoSQL数据库，甚至可以是字符文件或Excel文件。经过适当地处理和转化，开发接口软件，现有的各种数据（如专利数据、地理信息数据、电商数据、企业ERP系统数据），也可以转化成为EQL语言的数据源。

例如：事实是“鲍勃迪伦获得2016年诺贝尔文学奖，奖金8000000瑞典克朗”

这一事实在知识图谱系统中，以spo语句存放方式如下：

鲍勃迪伦:所获奖项:诺贝尔文学奖（日期:2016年，奖金:8000000瑞典克朗）

其中：

s=鲍勃迪伦,p=所获奖项,o=诺贝尔文学奖
q1=日期，v1=2016年
q2=奖金，v2=8000000瑞典克朗

在spo语句中，s,p,o分别是主语、谓语、宾语，例如“林肯:任职:美国总统”。或者是实体、属性、属性值，例如“地球:半径:6371公里”。

在知识图谱中，所有的名词都可以作为主语充当s,统称为实体。实体可以是万事万物，可以是具体的事物，例如某一个具体的人物、动物、植物、天体、山脉等，也可以是抽象的事物，如时间、空间、概念、事件、属性、官职、公式、定理等。

同一实体（如“萧伯纳”）的所有事实，组合在一起，构成这一实体的知识卡片。点击知识卡片编号的链接，用户可以调出知识卡片供查看，可以下载PDF或Word、Excel形式的知识卡片。对一组知识卡片，可以选择这些实体的若干个属性，生成Excel文件供下载。

可以充当p的实体，是谓语动词或实体的属性，统称为属性。例如“林肯:任职:美国总统”中的“任职”，或“地球:半径:6371公里”中的“半径”。

关于实体的编码方式，属性的编码建议以字母p开头，p代表属性property。其它所有实体的编码，建议以字母e开头，e代表实体entity。

宾语或属性值可以是一个实体，例如“美国总统”，也可以是具体的数据，例如“北京:面积:16410平方公里”中的“16410平方公里”。

一句spo语句代表一条事实，s:p:o是这一事实的主体描述部分，称为spo子句。(q1:v1,q2:v2,q3:v3,....qn:vn)部分称为修饰语，简称qv子句。修饰语说明了这一事实的细节部分。q1:v1或qn:vn称为一个修饰语，每个事实可以有0-n个修饰语(n为自然数）。

当n为0时，spo语句呈现为s:p:o的简单模式，对应的EQL语句可以是

?:p:o，　例如“?:首都:北京”，　答案：中国
s:?:o，　例如“鲍勃迪伦:?:男”，答案：性别
s:p:?，　例如“鸟:善于:?”，　　答案：飞行

以上的EQL语句分别用于查询主语、谓语、宾语，或者实体、属性、属性值，如此简单的用法，相信幼儿园小朋友都可以学会。

这种模式下的EQL语句，还可以进一步精简为：

?po，　例如“?首都北京”，　答案：中国
s?o，　例如“鲍勃迪伦?男”，答案：性别
sp?，　例如“鸟善于?”，　　答案：飞行

现在读者可以体会到EQL为什么称为极简查询语言(Extremely Simple Query Language)了。

基于人工智能的自然语言理解，EQL可以实现po和sp的自动分词，自动将“鸟善于?”转化为“鸟:善于:?”的s:p:o标准形式。

EQL语法的基本形式，就是对spo语句s:p:o(q1:v1,q2:v2,q3:v3,.....qn:vn)中的任一部分，用?代替，通过精准搜索知识图谱，返回计算结果。

举例如下:（支持本文查询的部分数据，请参阅附件2）

案例2　对spo语句中的实体s提问
　问题:谁获得了1925年的诺贝尔文学奖？
　EQL: ?:所获奖项:诺贝尔文学奖（日期:1925年）
　ANS: 萧伯纳 ehm001001

注：ehm001001是“萧伯纳”在知识图谱系统中的编号，点击这个编号可以显示关于“萧伯纳”的知识卡片，提供类似维基百科那样的丰富信息。

案例3　对spo语句中的属性p提问
　问题：萧伯纳和都柏林是什么关系？
　EQL:萧伯纳:？:都柏林
　ANS:出生地 p01000100

注：这样的查询可以帮助用户方便地用一个例子查出“属性”的名称

案例4 对spo语句中的属性值o提问
　问题：萧伯纳的出生地点在哪里？
　EQL:萧伯纳:出生地点:?
　ANS:都柏林 ep1900101

注：点击ep1900101,会发现这个都柏林，是爱尔兰首都都柏林，而不是美国加州旧金山湾区东部的小城都柏林（Dublin CA)。地名和人名很多时候会有同名的情况，在返回结果中给出编码，保证了EQL回答问题的精准性，这是采用EQL的搜索称为精准搜索的原因之一。当然，这样的精准搜索，要基于建立知识图谱时，就实现了对数据的准确语义理解和智能概念比对，在数据清洗和数据导入时要借助人工智能NLP自然语言理解和深度学习的许多尖端技术。

细心的读者可能已经发现，这句EQL中p用的是“出生地点”，而不是“出生地”，EQL可以执行出正确结果的原因是，在知识图谱中，已经存储了属性“出生地”的若干个“别名”。

对应的spo语句如下：

出生地:别名:出生地点

出生地:别名:出生城市
出生地:别名:BirthCity
出生地:别名:BirthPlace
用任何一个别名来构造EQL语句，都可以得到正确的结果。

案例5 对spo语句中的修饰语qv子句的部分细节进行查询
问题：萧伯纳获诺贝尔文学奖，奖金是多少？
EQL:萧伯纳:所获奖项:诺贝尔文学奖（奖金:？）
ANS:118165瑞典克朗

注：这句EQL语句，如果采用SQL数据库查询语言来写，根据数据库结构设计的不同，会有多种写法，以下是一个可能的版本。

```
SQL:select prizeAmount from person
        where personName = '萧伯纳'
         and prize = '诺贝尔文学奖'
```

通常情况下用户要写出正确的SQL语句，需要了解数据库中很多表的结构和每个字段的名字。如果要实现百科知识数据库，数据库中除了person表，还会有country、organization、place、event、product、animal等等许多表，很快就会超出普通人的记忆能力范围。

案例6 对spo语句中的修饰语qv子句的整体情况进行查询
问题：萧伯纳获得奥斯卡最佳改编剧本奖的情况？
EQL: 萧伯纳:所获奖项:奥斯卡最佳改编剧本奖(?)
ANS:（得奖作品：卖花女，
日期：1939年，
相关项：第11届奥斯卡金像奖）

注：这种情况下，EQL返回的结果是完整的修饰语qv子句，包含了全部细节。

案例7 得到多行的查询结果
问题：都有谁获得过诺贝尔文学奖?
EQL: ？:所获奖项:诺贝尔文学奖
ANS: 萧伯纳 ehm001001,
鲍勃迪伦 ehm001002,
石黑一雄 ehm001003
.....

注：当返回结果内容较多时，EQL先返回50行数据，用户可以继续点击得到所有数据或下50行数据。先返回50行数据,是为了让用户得到快速响应。

3.EQL的高级语法

案例8 得到多行的查询结果，并进行排序

问题：都有谁获得过诺贝尔文学奖?请按获奖时间倒序排序

EQL: ?x:所获奖项:诺贝尔文学奖(日期:?y)
\order by ?y desc

ANS: x=彼得·汉德克 ehm001009 y=2019年,
x=奥尔嘉·朵卡萩 ehm001008 y=2018年,
x=石黑一雄 ehm001003 y=2017年,
x=鲍勃迪伦 ehm001002 y=2016年,

.....

注：可以用?x或?x、？y、？z代表需要查询的变量。变量再增加时，可以用x,y,z1,z2,z3,....来代表，原则上变量总数不超过30个。

上面的EQL语句中“\order by ?y desc”称为排序子句，desc代表降序，EQL语言可以自动识别知识图谱内容中的数字和单位，例如“2019年”可以自动分析为数字“2019”和时间单位“年”。“日期”等时间属性的内容排序时，desc降序意味着把后发生的内容排在前面。

asc代表升序，如果未指定顺序时，按升序排列。

书写EQL的排序子句时，推荐另起一行，并进行缩进，以利于阅读理解。

案例9 得到多行的查询结果，并进行分组

问题：都有谁获得过诺贝尔文学奖?请按获奖者国籍分组，各国获奖者按时间先后次序排列

EQL: ?x:所获奖项:诺贝尔文学奖(日期:?y)
\group by ?x.国籍
\order by ?y asc

ANS: group x.国籍=美国
.....
x=尤金·奥尼尔 ehm001021 y=1936年,
x=赛珍珠 ehm001022 y=1938年,
x=威廉·福克纳 ehm001023 y=1949年,
x=欧内斯特·海明威 ehm001024 y=1954年,
.....
group x.国籍=法国
x=罗曼·罗兰 ehm001025 y=1915年,
x=阿那托尔·法朗士 ehm001026 y=1921年,
x=安德烈·纪德 ehm001027 y=1947年,
.....

注：“x.国籍”是spo语句中s.p表达式的一个例子，s.p称为属性表达式，表示实体s的属性p。例如当“x=欧内斯特·海明威”时，x.国籍=美国。

当s.p的结果是一个实体时，可以继续串联计算新的属性。这样形成的属性表达式为s.p1.p2.p3....pn，pn最大不超过p8

例如当“x=欧内斯特·海明威”时，x.国籍.首都=华盛顿

当返回数据量超过10万时，排序和分组计算量较大，可能会比较耗时，可以提示用户计算的进展情况和预计剩余时间。

案例10 得到多行的查询结果，并用Filter进行筛选

问题：1940年后，美国有谁获得过诺贝尔文学奖?

```
EQL: ?x:所获奖项:诺贝尔文学奖(日期:?y)
         \filter ?x.国籍=美国
         \filter ?y>1940年
ANS:  x=威廉·福克纳          ehm001023   y=1949年,
      x=欧内斯特·海明威      ehm001024   y=1954年,
      ......
```

注：\filter开头的子句称为Filter筛选子句，可以有多条筛选子句，它们之间是逻辑与的关系，也就是第一条筛选子句得到的结果，要交给第二条子句继续筛选，依次类推。

“\filter ?y>1940年”是正确的筛选子句，EQL会自动地分析“1940年”，提取其中的1940，用于大小比较的计算。

EQL中的比较运算符包括：

=	等于
!=	不等于
>	大于
>=	大于等于
<	小于
<=	小于等于

案例11 得到多行的查询结果，并用字符串模式匹配进行筛选

问题：诺贝尔文学奖得主中，有名叫“威廉”的吗？哪年得的奖?

```
EQL:  ?x:所获奖项:诺贝尔文学奖(日期:?y)
          \filter ?x \match '%威廉%'
ANS:  x=威廉·福克纳          ehm001023   y=1949年
      x=威廉·戈尔丁          ehm001067   y=1983年
      ......
```

注：\match表示匹配字符串，后面跟的是字符串匹配模版，如'%威廉%'。%可以匹配0-n个字符，也就是任意的字符串或空字符串。如果要匹配“威廉”在开头或结尾的字符串，匹配模版可以写成'威廉%'或'%威廉'。

另一个字符串匹配符是'_','_'确定匹配一个字符，可以是英文字符，也可以是汉字字符，或者是任何unicode编码的字符。

例如，匹配模版'William G_lding'可以匹配'William Golding'。匹配模版'赵_龙'可以匹配到'赵子龙'、'赵小龙'等。

在字符串匹配模版中，只使用两个匹配符号'%'和'_'，再一次体现了EQL的极简设计理念。

在《道德经》中，老子写到：“道生一，一生二，二生三，三生万物。”在字符串匹配的小世界中，'%'和'_'就是道和一。宇宙的本质也是道和一,愿道与你合一，原力与你同在。

如果要匹配本身含'%'或者'_'的字符串，可以用'\%'代表'%'，用'_'代表'_'。

例如：

```
EQL：?x:性质:歌曲
        \fileter ?x match '%99\%合格%'
ANS: x=《99%合格男朋友》 e03301003
```

类似地，如果spo语句的内容中有':'或'?',可以用\:代表':',用\?代表'?'。例如，有两本书分别是《生命是什么?--活细胞的物理观》和《中亚:马背上的文化》，查询作者时，可以这么写EQL语句：

```
EQL：《生命是什么\?--活细胞的物理观》:作者:?
ANS: 埃尔温·薛定谔  ehm001097

EQL：《中亚\:马背上的文化》:作者:?
ANS: 项英杰  ehm021038
```

为方便用户输入，EQL语句中的？和：，可以用英文输入法输入,也可以用中文输入法输入。

案例12 采用\and、\or、\not等实现与或非逻辑运算

```
问题：谁同时获得了诺贝尔奖和奥斯卡金像奖？
EQL:（？x:所获奖项）:性质:诺贝尔奖
        \and
     (？x:所获奖项）:性质:奥斯卡金像奖
ANS: 萧伯纳    ehm001001,
     鲍勃迪伦 ehm001002
```

注：

a.EQL可以采用\and、\or、\not来进行逻辑与、或、非的计算，\true和\false表示逻辑的真与假。

b.(?x:所获奖项)这样的形式，代表用spo语句中的s和p项来计算，（s:p)计算后得到所有满足(s:p:o)条件的o,可以作为后续的语句的主语，继续参与查询。

c. 简单描述一下对某一行spo语句数据的逻辑计算过程

假设，当x=萧伯纳时，其中一行spo事实数据为

萧伯纳:所获奖项:诺贝尔文学奖

（日期：1925年，

奖金：118165瑞典克朗）

基于这一行数据进行查询的计算过程如下：

（？x：所获奖项）：性质：诺贝尔奖

=（萧伯纳：所获奖项）：性质：诺贝尔奖

= 诺贝尔文学奖：性质：诺贝尔奖

= \true

计算的最后一步得到\true，是因为知识图谱中有以下spo语句数据支持：

诺贝尔文学奖:性质:诺贝尔奖

这一行计算得到的逻辑“\true”继续参与后续的逻辑计算，最后通过查询知识图谱中的相关数据，计算出“x=萧伯纳”和“x=鲍勃迪伦”是满足查询条件的结果

d.在Wikidata知识图谱中，用来做同样查询的SPAQL语句如下：

```
#People that received both Academy Award and Nobel Prize
SELECT DISTINCT ?person ?personLabel WHERE {
    ?person wdt:P166/wdt:P31? wd:Q7191 .
    ?person wdt:P166/wdt:P31? wd:Q19020 .
  SERVICE wikibase:label {
    bd:serviceParam wikibase:language "[AUTO_LANGUAGE],en" .
   }
 }
```

要读懂上面的SPARQL语句，需要了解在Wikidata中，P166代表所获奖项，P31代表性质（instance of)，Q7191代表诺贝尔奖，Q19020代表奥斯卡金像奖。可以看出EQL语言明显比SPARQL更简明易懂。

Wikidata中的SPARQL查询，实现了精准搜索，但牺牲了用户学习和使用的方便性。而EQL既能做到方便易学，又能保持搜索的精准性。

案例13 采用“\and”代表逻辑与的原因

问题：谁是《Pride and Prejudice》和《Sense and Sensibility》这两本书的作者？

EQL: 《Pride and Prejudice》:作者:?x

\and

《Sense and Sensibility》:作者:?x

ANS: x=简·奥斯汀 ehf001035

注：在上面的EQL语句中，有三个出现and的地方，\and代表逻辑运算一目了然。

EQL用“\and”而不是“and”或“&”代表逻辑与计算的原因，是因为在spo语句中，很多内容会有“and”或“&”出现。采用“\and”代表逻辑与，可以大大减少逻辑歧义。

EQL采用\or,\not,\true,\false作为逻辑和语义符号的原因，与\and的情况相似。

案例14 人名或地名等实体重名情况的处理

问题：哪些作家出生在都柏林？

EQL: ？x:职业:作家
\and
？x:出生地:都柏林

ANS: 萧伯纳 ehm001001,
詹姆斯·乔伊斯 ehm001088,
.....

EQL重名提示：
1.都柏林 都柏林_爱尔兰 缺省值
2.都柏林 都柏林_加州
以上查询结果按缺省值“都柏林=都柏林_爱尔兰”执行获取

注：以上的“ EQL重名提示：
1.都柏林 都柏林_爱尔兰 缺省值
2.都柏林 都柏林_加州 .....”
也是EQL查询结果的一部分。

人名或地名经常有重名的情况，如果其中一个是明显更知名，我们可以将其认为是缺省的选项。例如，都柏林通常指爱尔兰首都都柏林，我们在特指“加州的都柏林”时，采用“都柏林_加州”的形式。地名的重名情况，我们通常在后面加上其所属更高一级的行政区域来区分。

人名我们通常用职业来区分，例如：
1.姚明 姚明_篮球运动员 缺省值
2.姚明 姚明_作曲家
如果查询篮球运动员姚明的情况，我们直接用“姚明”来表示，如：
EQL:(姚明:妻子):毕业学校:?
如果查询作曲家姚明的情况，我们用“姚明_作曲家”来表示，如：
EQL: 姚明_作曲家:出生地:?

在人名和职业都相同的情况下，我们可以继续用更多的内容来区分，如：
1.宋佳 宋佳_演员_80后女演员 缺省值 别名:小宋佳
2.宋佳 宋佳_演员_60后女演员

3.宋佳　宋佳_排球运动员

如果查询80后女演员宋佳的情况，我们可以用“宋佳”或“小宋佳”来表示，因重名情况较多，推荐用“小宋佳”来查询，以方便其它用户理解你的EQL语句的真实含义，如：

EQL：小宋佳:主要作品:?

案例15 用EQL语句计算统计结果,使用count函数

问题：谁同时获得了诺贝尔奖和奥斯卡奖？请统计人数

EQL：(?x:所获奖项):性质:诺贝尔奖
\and
(?x:所获奖项):性质:奥斯卡金像奖,
?y=count(?x)

ANS:x=萧伯纳　ehm001001,
鲍勃迪伦 ehm001002
y=2

案例16 用EQL语句计算统计结果,使用avg,max函数

问题：谁同时获得了诺贝尔奖和奥斯卡奖？
请统计他们的诺贝尔奖金的平均数和最高值

EQL：(?x:所获奖项):性质:诺贝尔奖
\and
(?x:所获奖项):性质:奥斯卡金像奖,
?x:所获奖项:?y(奖金:?z1)
\and
?y:性质:诺贝尔奖,
? z2=avg(?z1),
? z3=max(?z1)

ANS：x=萧伯纳　ehm001001,
鲍勃迪伦 ehm001002
y=诺贝尔文学奖
z1=118165瑞典克朗(x=萧伯纳 ehm001001),
8000000瑞典克朗(x=鲍勃迪伦 ehm001002)
z2=4059082.5瑞典克朗
z3= 8000000瑞典克朗(x=鲍勃迪伦 ehm001002)

EQL语言中的统计函数包括：count(计数），avg(平均数），sum(总计），max(取最大值），min(取最小值)。统计的范围由统计函数括号内变量所在的spo语句及其之前的spo语句确定。

例如，本案例中avg(?z1)统计的范围，由这部分EQL语句确定：

(?x:所获奖项):性质:诺贝尔奖
\and
(?x:所获奖项):性质:奥斯卡金像奖,

?x:所获奖项:?y(奖金:?z1)
\and
?y:性质:诺贝尔奖,
先找到满足以上EQL语句条件的?x、?y、?z1，再用这些?z1进行平均值avg(?z1)的统计。

这些统计函数通常直接跟在“=”或“(”后面，如“？z2=avg(?z1)”，因此不必写成“\count”或“\avg”这样的形式。

EQL语言用“，”将可执行的EQL语句分开，总是先执行前面的EQL语句，从第一句开始依次执行，同时，前面语句中用到的变量如?x,?y的结果，在后面的语句中保留不变。变量采用的顺序应为?x、？y、？z或？x、？y、？z1、？z2、？z3，...，?zn，依次类推，变量最多不超过50个。

案例17 查询中采用别名
问题：G.B. Shaw出生地在哪里？
EQL: G.B.Shaw:出生地点：？
ANS: 都柏林 ep1900101

注：执行时，作为别名的“G.B Shaw”自动替换成“萧伯纳”
作为别名的“出生地点”自动替换成“出生地”
EQL语言中，当采用别名查询时，结果不变。

案例18 模糊查询
当未找到适当的别名时，EQL可以利用NLP自然语言理解能力，自动匹配语义最接近（或文字最相似）的内容，提示给用户。
问题：G.E.Shaw出生地在哪里？
（假设用户记错了萧伯纳的英文名，或者输入错误）
EQL: G.E.Shaw:出生地:?

系统提示：未查到“G.E.Shaw”相关信息，你实际想查询的是“G.B.Shaw(萧伯纳)：出生地：？”吗？（y/n)
用户输入y后，系统给出正确的查询结果

EQL:G.B.Shaw:出生地:?
ANS: 都柏林 ep1900101

案例19 EQL与Lambda演算的对应关系
问题：找出有孩子出生在纽约的人
EQL:(?:childern):BirthPlace:New York
ANS:鲍勃迪伦 ehm001002,
麦尔维尔·阿瑟·费曼 ehm001359
.....

注：鲍勃迪伦有个孩子雅戈布·狄伦（Jakob Dylan)出生于纽约，也是个歌手。

麦尔维尔·阿瑟·费曼是著名物理学家费曼的父亲,费曼出生在纽约皇后区。麦尔维尔教会了理蒂（费曼小时候的昵称）怎样思考。他让理蒂设想他遇见了火星人，火星人肯定要问很多关于地球的问题。比如说，为什么地球人要在夜里睡觉呢？麦尔维尔真是个有爱心有才华的好父亲。

阿隆佐·邱奇(Alonzo Church)是计算机行业先驱图灵的博士导师，他于1936年引入的Lambda演算（λ-calculus）简洁而强大，被称为最小的通用程序设计语言，对数理逻辑和计算机的发展有巨大的推动作用，早期的人工智能语言Lisp最核心的基础就是Lambda演算。

绝大多数EQL语句都可以转化为Lambda演算表达式，这将为EQL语言提供数学和数理逻辑的强大支撑。

案例15的EQL语句可以转化为Lambda演算语句，对比如下：

EQL:　　(?:childern):BirthPlace:New York

λ演算：λx.∃y.Children(x, y) ∧ BirthPlace(y, NewYork)

对比之后可以看出，EQL比λ演算更加简洁易懂一些。

案例20 只需要返回部分结果变量时

问题：找出有10孩子以上的音乐家

```
EQL: ?x:职业:音乐家
       \and
     ?x:子女:?y
       \and
     (count(?y)>=10),
     ?z=count(?y),
     ANS ?x,?x.母语, ?z

ANS:x=约翰·塞巴斯蒂安·巴赫 ehm001077,
    x.母语=德语
    z=20
    .......
```

注：当只需要返回部分变量时，可以用ANS语句指定返回的变量，ANS是answer的缩写，也可以指定返回变量的某个属性，如“x.母语”。在本案例中ANS语句中没有?y,所以?y的结果将不被返回。

案例21　分步写出EQL语句

问题:找出美国面积最大的州

```
EQL: ?x:性质:美国州份,
     ?x:面积:?y,
     ?z=max(?y),
     ANS ?z
```

ANS: z=152万平方公里(x=阿拉斯加州)

注：再作一次EQL语句与Lambda演算语句的对比：

EQL: ?x:性质:美国州份,
?x:面积:?y,
?z=max(?y),
ANS ?z

λ演算：argmax(λx.InstanceOf(x,USState),λx.λy.Area(x, y))

相比起来，EQL语句可以分四步写出，更容易被普通人逐步掌握。λ演算更适合特别聪明的天才少年，可以一气呵成。

4.EQL的未来发展

案例22 概念推理查询(EQL语言实现时的可选项)

问题：萧伯纳的外公是谁？

EQL: 萧伯纳:外公:?

ANS: Walter Gurly ehm001059

注:在EQL中，可以采用一些已经定义好的概念推理关系进行查询。

假设我们的知识图谱中只存储了每个人的父母信息、子女信息和出生日期等，并没有直接存这些人外公的信息。我们可以通过概念推理的方式来进行直接查询。

系统中关于“外公”属性的定义如下：

x:外公:y
=(x:母亲):父亲:y

系统自动将原查询

EQL: 萧伯纳:外公:?

转换为以下形式的查询

EQL: (萧伯纳:母亲):父亲:?

从而得到正确的结果:

ANS: Walter Gurly ehm001059

类似地，我们可以定义爷爷、伯伯、叔叔、舅舅、姨妈、姑妈、哥哥、弟弟、姐姐、妹妹等概念的概念推理公式。

例如，可以如下定义“哥哥”属性：

x:哥哥:y
= ((x:父亲):子女:y
\and
(x:出生日期)>(y:出生日期)
\and
y:性别:男)
\or

((x:母亲）:子女:y
\and
(x:出生日期）>(y:出生日期）
\and
y:性别:男）

这么复杂的定义可以将x的同父异母哥哥和同母异父哥哥都包括在内，这样的概念推理计算量较大，因此实际实现时，建议保留计算好的结果，在知识图谱中，加入一条“x:哥哥:y"的spo事实语句。

概念推理查询如果全面实施，实现难度较大，可以作为EQL 1.0版的可选项。计划将此功能包含在EQL 2.0版中。

案例23 自然语言填空查询

EQL的未来发展目标之一，是实现自然语言填空操作，也就是说，把自然语言中的任意一个词换成“？”，可以自动填空，得到结果。

EQL填空版:萧伯纳的？是Walter Curly
ANS:外公

EQL填空版:萧伯纳获得诺贝尔文学奖，奖金？
ANS:118165瑞典克朗

注：相信这种填空功能，非常适合于使用类似于Google BERT这样的深度学习预训练模型，欢迎这方面的专家作出更多的研究开发。

对于作家和白领一族，写文章时可以非常方便地采用EQL填空版，帮助他们迅速找到写作所需的背景知识内容。

案例24 EQL语音版

EQL的另一种发展前景，是用户用语音直接提问，EQL实现语音识别，自动生成EQL语句，并且语音回复用户查询结果，这种模式可以用于实现加强版的智能音箱。

EQL语音提问:萧伯纳与Walter Curly是什么关系？
ANS语音回答:Walter Curly是萧伯纳的外公

案例25 对知识图谱的增加请求

EQL作为极简查询语言，通常不建议用户对知识图谱中的数据进行增删改操作。如果用户非常希望改变数据时，可以通过\suggest语句提出数据增删改

请求，知识图谱的管理员会对用户提交的数据进行核验，核验通过后才会改动，核验和提交的结果将告知用户。

问题：希望增加一条关于海明威主要作品的数据
EQL:\suggest add
欧内斯特·海明威:主要作品:《流动的圣节》
（首次出版:1964年，ISBN:0-684-82499-X)
\ref1:维基百科 www.wikipedia.com,
\ref2:《海明威研究文集》 ISBN 978-7-5447-3164-5
ANS: ‘\suggest add’请求已受理
‘\suggest add’请求已审核通过，已成功更新知识图谱

注：可以用\suggest add语句请求在知识图谱中增加数据，增加的数据以spo语句的形式呈现，在本案例中spo事实语句是:
欧内斯特·海明威:主要作品:《流动的圣节》
（首次出版:1964年，ISBN:0-684-82499-X)

通常，用户提交数据时，应提交数据的来源以便于审核。在本案例中，用户在提交了两个参考来源：维基百科和《海明威研究文集》，并提交了网站的地址和ISBN书号，最终成功通过了审核。

用户可以提交0-n条数据参考来源信息，用\ref，或\ref1,\ref2等来开始ref子句。

案例26 对知识图谱的更改请求
问题：希望更改一条关于海明威主要作品的数据
EQL:\suggest change
欧内斯特·海明威:主要作品:《流动的圣节》
（首次出版:1964年，ISBN:0-684-82499-X)
\changeTo
欧内斯特·海明威:主要作品:《流动的圣节》
（首次出版:1964年，ISBN:0-684-82499-X,
出版社：Simon & Schuster)
\ref1:维基百科 www.wikipedia.com,
\ref2:《海明威研究文集》 ISBN 978-7-5447-3164-5
ANS: ‘\suggest change’请求已受理
‘\suggest change’请求已审核通过，已成功更新知识图谱

注：可以用\suggest change语句请求在知识图谱中更改数据，先要列出更改前的spo语句，然后，在\changeTo后边放在更改后的完整spo语句，更改后的spo语句如有任何错误，此次更改请求将被拒绝。

案例27 对知识图谱的删除请求
问题：希望删除一条关于海明威主要作品的数据
EQL:\suggest delete
欧内斯特·海明威:主要作品:《流动的圣节》

ANS: ‘\suggest delete’请求已受理
‘\suggest change’请求已被拒绝

注：通常用户不应该提交删除请求，因为对某一用户无用的数据，可能对别的用户有用。提交删除请求时，只要提交s:p:o主体部分即可，不用提交qv修饰子句。在此案例中，客户的删除请求被拒绝。

每个删除请求如果通过时，最多删去一条spo事实语句。

5.总结

EQL，又称极简查询语言(Extremely Simple Query Language)，可以普遍用于知识图谱、精准搜索、强人工智能、数据库、智能音箱、专利检索等领域。从本文案例上可以看出，EQL语言在设计上采用极简原则，明显比SQL、SPARQL等查询语言更简单明了，更易于被普通人掌握。

EQL语言与λ演算的互相转化，揭示了EQL语言的数学本质，为EQL语言的严密性和逻辑完整性奠定了坚实的基础。

EQL语言对应的数据底层存储系统，可以有多种选择，如SQL数据库、RDF数据库、图数据库、MongoDB等NoSQL数据库，甚至可以是字符文件或Excel文件。经过适当地处理和转化，开发接口软件，现有的各种数据（如专利数据、地理信息数据、电商数据、企业ERP系统数据），也可以抽象转化成为EQL语言的数据源。在应用方面，相信EQL可以方便地应用到银行、电信、医疗、互联网、制造、服务业等各个行业。

我们目前已完成EQL语言的设计，将配合世界常识知识图谱系统推出EQL的实现版，欢迎各界朋友的投资和赞助。我们实现的世界常识知识图谱系统将覆盖人类绝大多数学科的绝大多数概念、名词、常识和公式，从中英双语开始，最终将支持世界所有国家和所有民族的语言。预计这个知识图谱中会几亿个名词实体，有几百亿条事实数据。

在底层实现上，EQL将通过采用人工智能中的NLP自然语言理解和深度学习等技术，以及大数据和搜索领域的技术，确保实现高速精准的信息搜索，从而帮助用户更好地挖掘出数据资产中蕴含的财富。同时，我们欢迎有志之士，独立设计实现EQL基于各种底层存储系统的接口，特别是SQL数据库、Excel和文件系统的接口。我们保留EQL的版权和商业使用权，欢迎朋友们联系商业上的使用授权。

本文第一作者刘韩先生的使命就是发展强人工智能，我发愿终生致力于强人工智能的研究和开发，致力于整合灵性、科学、艺术、商业、AI等多学科的智慧，创立整合这些学科的合一之道---“钻石思维模型”，EQL是刘韩先生（微信号:lh7648)基于“钻石思维模型”创造的第一个精神产品。

EQL语言和完善的世界常识知识图谱系统，可以共同构成未来强人工智能的基础，弥补目前人工智能系统对世界常识缺少理解引发的缺憾。EQL语言不仅可以供人类使用，也可以作为机器人之间的数据查询和数据交换的基本语言。强人工智能的研究涉及数学、计算机科学、脑科学、语言学、心理学、经济学、社会学等诸多学科，大大超出了刘韩先生的能力范围圈，期待各方面的

朋友多多赐教，多多合作，为帮助更多的普通人释放潜能，实现理想，为人类的解放共同努力。

参考文献

Alan Turing (1950) Computing Machinery and Intelligence. Mind 49: 433-460.

Alonzo Church An Unsolvable Problem of Elementary Number Theory American Journal of Mathematics, Vol. 58, No. 2. (Apr., 1936), pp. 345-363.

Donald D. Chamberlin and Raymond F. Boyce. SEQUEL: A Structured English Query Language

S. Harris and A. Seaborne. SPARQL 1.1 query language. In W3C Working Draft, 12 May, 2011.

Denny Vrandecic and Markus Krotzsch. Wikidata: A Free Collaborative Knowledge Base

Percy Liang Lambda Dependency-Based Compositional Semantics

刘韩 《人工智能简史》 人民邮电出版社

附录2

支持本论文EQL查询的部分事实数据，以下每一项是一条spo事实语句：

1.萧伯纳：所获奖项：诺贝尔文学奖
（日期：1925年，奖金：118165瑞典克朗）
2.萧伯纳：所获奖项：奥斯卡最佳改编剧本奖
（得奖作品：卖花女，
日期：1939年，
相关项：第11届奥斯卡金像奖）
3.诺贝尔文学奖：性质：诺贝尔奖
4.诺贝尔文学奖：性质：文学奖
5.鲍勃迪伦：所获奖项：诺贝尔文学奖
（日期：2016年，
奖金：8000000瑞典克朗，
下一项：石黑一雄）
6.鲍勃迪伦：所获奖项：奥斯卡最佳原创歌曲奖
（日期：2000年，
得奖作品：Things Have Changed)
7.奥斯卡最佳原创歌曲奖：性质：奥斯卡金像奖
8.奥斯卡最佳改编剧本奖：性质：奥斯卡金像奖
9.萧伯纳：英文名：George Bernard Shaw
10.鲍勃迪伦：英文名：Bob Dylan
11.萧伯纳：出生地：都柏林
12.萧伯纳：别名：G.B.Shaw
13.萧伯纳:别名：Bernard Shaw
14.出生地：别名：出生地点
15.奥斯卡金像奖：别名：奥斯卡奖