# Kylin安装

## 安装指南

### 软件要求

* Hadoop: 2.7+, 3.1+ (since v2.5)
* Hive: 0.13 - 1.2.1+
* HBase: 1.1+, 2.0 (since v2.5)
* Spark (可选) 2.3.0+
* Kafka (可选) 1.0.0+ (since v2.5)
* JDK: 1.8+ (since v2.5)
* OS: Linux only, CentOS 6.5+ or Ubuntu 16.0.4+

在 Hortonworks HDP 2.2-2.6 and 3.0, Cloudera CDH 5.7-5.11 and 6.0, AWS EMR 5.7-5.10, Azure HDInsight 3.5-3.6 上测试通过。

我们建议您使用集成的 sandbox 来试用 Kylin 或进行开发，比如 [HDP sandbox](http://hortonworks.com/products/hortonworks-sandbox/)，且要保证其有至少 10 GB 内存。在配置沙箱时，我们推荐您使用 Bridged Adapter 模型替代 NAT 模型。

### 硬件要求

运行 Kylin 的服务器的最低配置为 4 core CPU，16 GB 内存和 100 GB 磁盘。 对于高负载的场景，建议使用 24 core CPU，64 GB 内存或更高的配置。

### Hadoop 环境

Kylin 依赖于 Hadoop 集群处理大量的数据集。您需要准备一个配置好 HDFS，YARN，MapReduce，Hive， HBase，Zookeeper 和其他服务的 Hadoop 集群供 Kylin 运行。  
Kylin 可以在 Hadoop 集群的任意节点上启动。方便起见，您可以在 master 节点上运行 Kylin。但为了更好的稳定性，我们建议您将 Kylin 部署在一个干净的 Hadoop client 节点上，该节点上 Hive，HBase，HDFS 等命令行已安装好且 client 配置（如 core-site.xml，hive-site.xml，hbase-site.xml及其他）也已经合理的配置且其可以自动和其它节点同步。

运行 Kylin 的 Linux 账户要有访问 Hadoop 集群的权限，包括创建/写入 HDFS 文件夹，Hive 表， HBase 表和提交 MapReduce 任务的权限。

### Kylin 安装

从 [Apache Kylin下载网站](https://kylin.apache.org/download/) 下载一个适用于您 Hadoop 版本的二进制文件。例如，适用于 HBase 1.x 的 Kylin 2.5.0 可通过如下命令行下载得到：

cd /usr/local/

wget http://mirror.bit.edu.cn/apache/kylin/apache-kylin-2.5.0/apache-kylin-2.5.0-bin-hbase1x.tar.gz

1.解压 tar 包，配置环境变量 $KYLIN\_HOME 指向 Kylin 文件夹。

tar -zxvf apache-kylin-2.5.0-bin-hbase1x.tar.gz

cd apache-kylin-2.5.0-bin-hbase1x

export KYLIN\_HOME=`pwd`

从 v2.6.1 开始， Kylin 不再包含 Spark 二进制包; 您需要另外下载 Spark，然后设置SPARK\_HOME系统变量到 Spark 安装目录：

export SPARK\_HOME=/path/to/spark

或者使用脚本下载:

$KYLIN\_HOME/bin/download-spark.sh

### Kylin tarball 目录

* bin: shell 脚本，用于启动／停止 Kylin，备份／恢复 Kylin 元数据，以及一些检查端口、获取 Hive/HBase 依赖的方法等；
* conf: Hadoop 任务的 XML 配置文件，这些文件的作用可参考[配置页面](http://kylin.apache.org/docs/install/configuration.html)
* lib: 供外面应用使用的 jar 文件，例如 Hadoop 任务 jar, JDBC 驱动, HBase coprocessor 等.
* meta\_backups: 执行 bin/metastore.sh backup 后的默认的备份目录;
* sample\_cube 用于创建样例 Cube 和表的文件。
* spark: 自带的 spark。
* tomcat: 自带的 tomcat，用于启动 Kylin 服务。
* tool: 用于执行一些命令行的jar文件。

### 检查运行环境

Kylin 运行在 Hadoop 集群上，对各个组件的版本、访问权限及 CLASSPATH 等都有一定的要求，为了避免遇到各种环境问题，您可以运行 $KYLIN\_HOME/bin/check-env.sh 脚本来进行环境检测，如果您的环境存在任何的问题，脚本将打印出详细报错信息。如果没有报错信息，代表您的环境适合 Kylin 运行。

### 启动 Kylin

运行 $KYLIN\_HOME/bin/kylin.sh start 脚本来启动 Kylin，界面输出如下：

Retrieving hadoop conf dir...

KYLIN\_HOME is set to /usr/local/apache-kylin-2.5.0-bin-hbase1x

......

A new Kylin instance is started by root. To stop it, run 'kylin.sh stop'

Check the log at /usr/local/apache-kylin-2.5.0-bin-hbase1x/logs/kylin.log

Web UI is at http://<hostname>:7070/kylin

### 使用 Kylin

Kylin 启动后您可以通过浏览器 http://<hostname>:7070/kylin 进行访问。其中<hostname>为具体的机器名、IP 地址或域名，默认端口为 7070。初始用户名和密码是ADMIN/KYLIN。  
服务器启动后，您可以通过查看 $KYLIN\_HOME/logs/kylin.log 获得运行时日志。

### 停止 Kylin

运行 $KYLIN\_HOME/bin/kylin.sh stop 脚本来停止 Kylin，界面输出如下：

Retrieving hadoop conf dir...

KYLIN\_HOME is set to /usr/local/apache-kylin-2.5.0-bin-hbase1x

Stopping Kylin: 25964

Stopping in progress. Will check after 2 secs again...

Kylin with pid 25964 has been stopped.

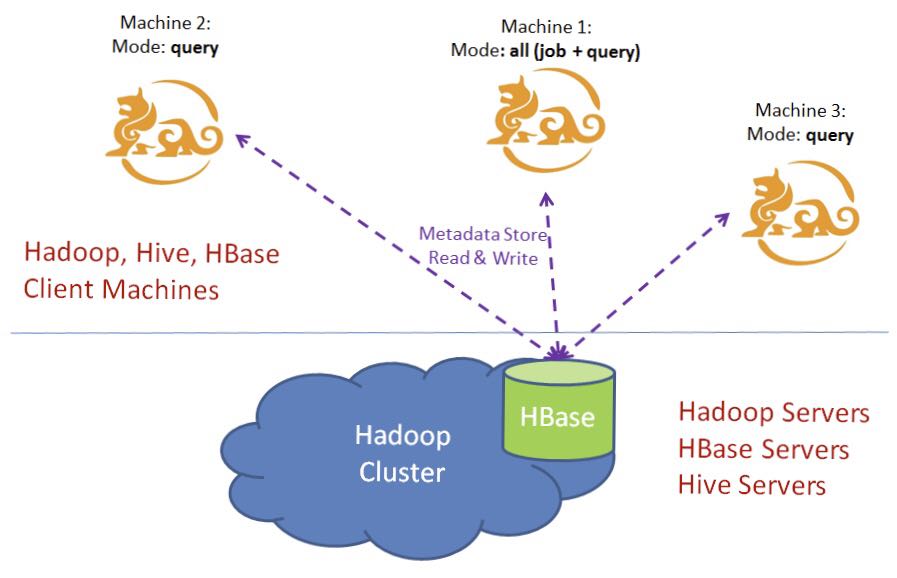
您可以运行 ps -ef | grep kylin 来查看 Kylin 进程是否已停止。

### HDFS 目录结构

Kylin 会在 HDFS 上生成文件，根目录是 “/kylin/”, 然后会使用 Kylin 集群的元数据表名作为第二层目录名，默认为 “kylin\_metadata” (可以在conf/kylin.properties中定制).

通常, /kylin/kylin\_metadata 目录下会有这么几种子目录cardinality,coprocessor,kylin-job\_id,resources,jdbc-resources.  
1. cardinality: Kylin 加载 Hive 表时，会启动一个 MR 任务来计算各个列的基数，输出结果会暂存在此目录。此目录可以安全清除。  
2. coprocessor: Kylin 用于存放 HBase coprocessor jar 的目录；请勿删除。  
3. kylin-job\_id: Cube 计算过程的数据存储目录，请勿删除。 如需要清理，请遵循 [storage cleanup guide](http://kylin.apache.org/docs/howto/howto_cleanup_storage.html).  
4. resources: Kylin 默认会将元数据存放在 HBase，但对于太大的文件（如字典或快照），会转存到 HDFS 的该目录下，请勿删除。如需要清理，请遵循 [cleanup resources from metadata](http://kylin.apache.org/docs/howto/howto_backup_metadata.html)  
5. jdbc-resources：性质同上，只在使用 MySQL 做元数据存储时候出现。

## 集群模式部署

Kylin 实例是无状态的服务，运行时的状态信息存储在 HBase metastore 中。 出于负载均衡的考虑，您可以启用多个共享一个 metastore 的 Kylin 实例，使得各个节点分担查询压力且互为备份，从而提高服务的可用性。下图描绘了 Kylin 集群模式部署的一个典型场景：  


### Kylin 集群模式部署

如果您需要将多个 Kylin 节点组成集群，请确保他们使用同一个 Hadoop 集群、HBase 集群。然后在每个节点的配置文件 $KYLIN\_HOME/conf/kylin.properties 中执行下述操作：

1. 配置相同的 kylin.metadata.url 值，即配置所有的 Kylin 节点使用同一个 HBase metastore。
2. 配置 Kylin 节点列表 kylin.server.cluster-servers，包括所有节点（包括当前节点），当事件变化时，接收变化的节点需要通知其他所有节点（包括当前节点）。
3. 配置 Kylin 节点的运行模式 kylin.server.mode，参数值可选 all, job, query 中的一个，默认值为 all。  
   job 模式代表该服务仅用于任务调度，不用于查询；query 模式代表该服务仅用于查询，不用于构建任务的调度；all 模式代表该服务同时用于任务调度和 SQL 查询。

**注意：**默认情况下只有**一个实例**用于构建任务的调度 （即 kylin.server.mode 设置为 all 或者 job 模式）。

### 任务引擎高可用

从 v2.0 开始, Kylin 支持多个任务引擎一起运行，相比于默认单任务引擎的配置，多引擎可以保证任务构建的高可用。

使用多任务引擎，你可以在多个 Kylin 节点上配置它的角色为 job 或 all。为了避免它们之间产生竞争，需要启用分布式任务锁，请在 kylin.properties 里配置：

kylin.job.scheduler.default=2

kylin.job.lock=org.apache.kylin.storage.hbase.util.ZookeeperJobLock

并记得将所有任务和查询节点的地址注册到 kylin.server.cluster-servers。

### 安装负载均衡器

为了将查询请求发送给集群而非单个节点，您可以部署一个负载均衡器，如[Nginx](http://nginx.org/en/)，[F5](https://www.f5.com/)或[cloudlb](https://rubygems.org/gems/cloudlb/)等，使得客户端和负载均衡器通信代替和特定的 Kylin 实例通信。

### 读写分离部署

为了达到更好的稳定性和最佳的性能，建议进行读写分离部署，将 Kylin 部署在两个集群上，如下：

* 一个 Hadoop 集群用作 **Cube 构建**，这个集群可以是一个大的、与其它应用共享的集群；
* 一个 HBase 集群用作 **SQL 查询**，通常这个集群是专门为 Kylin 配置的，节点数不用像 Hadoop 集群那么多，HBase 的配置可以针对 Kylin Cube 只读的特性而进行优化。

这种部署策略是适合生产环境的最佳部署方案，关于如何进行读写分离部署，请参考 [Deploy Apache Kylin with Standalone HBase Cluster](http://kylin.apache.org/blog/2016/06/10/standalone-hbase-cluster/)。

## Kylin配置

<http://kylin.apache.org/cn/docs/install/configuration.html>

# Kylin教程

## 样例Cube快速入门

Kylin 提供了一个创建样例 Cube 脚本；脚本会创建五个样例 Hive 表:

1. 运行 ${KYLIN\_HOME}/bin/sample.sh；重启 Kylin 服务器刷新缓存;
2. 用默认的用户名和密码 ADMIN/KYLIN 登陆 Kylin 网站，选择 project 下拉框（左上角）中的 learn\_kylin 工程;
3. 选择名为 kylin\_sales\_cube 的样例 Cube，点击 “Actions” -> “Build”，选择一个在 2014-01-01 之后的日期（覆盖所有的 10000 样例记录);
4. 点击 “Monitor” 标签，查看 build 进度直至 100%;
5. 点击 “Insight” 标签，执行 SQLs，例如:

select part\_dt, sum(price) as total\_sold, count(distinct seller\_id) as sellers from kylin\_sales group by part\_dt order by part\_dt

1. 您可以验证查询结果且与 Hive 的响应时间进行比较;

**Streaming 样例 Cube 快速入门**

Kylin 也提供了 streaming 样例 Cube 脚本。该脚本将会创建 Kafka topic 且不断的向生成的 topic 发送随机 messages。

1. 首先设置 KAFKA\_HOME，然后启动 Kylin。
2. 运行 ${KYLIN\_HOME}/bin/sample.sh，它会在 learn\_kylin 工程中生成 DEFAULT.KYLIN\_STREAMING\_TABLE 表，kylin\_streaming\_model 模型和 kylin\_streaming\_cube Cube。
3. 运行 ${KYLIN\_HOME}/bin/sample-streaming.sh，它会在 localhost:9092 broker 中创建名为 kylin\_streaming\_topic 的 Kafka Topic。它也会每秒随机发送 100 条 messages 到 kylin\_streaming\_topic。
4. 遵循标准 Cube build 过程，并触发名为 kylin\_streaming\_cube 的 Cube 的构建。
5. 点击 “Monitor” 标签，查看 build 进度。直至至少有一个 job 达到 100%。
6. 点击 “Insight” 标签，执行 SQLs，例如:

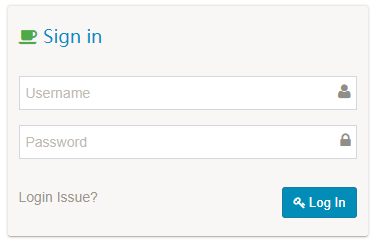
select count(\*), HOUR\_START from kylin\_streaming\_table group by HOUR\_START

1. 验证查询结果。

## WEB界面

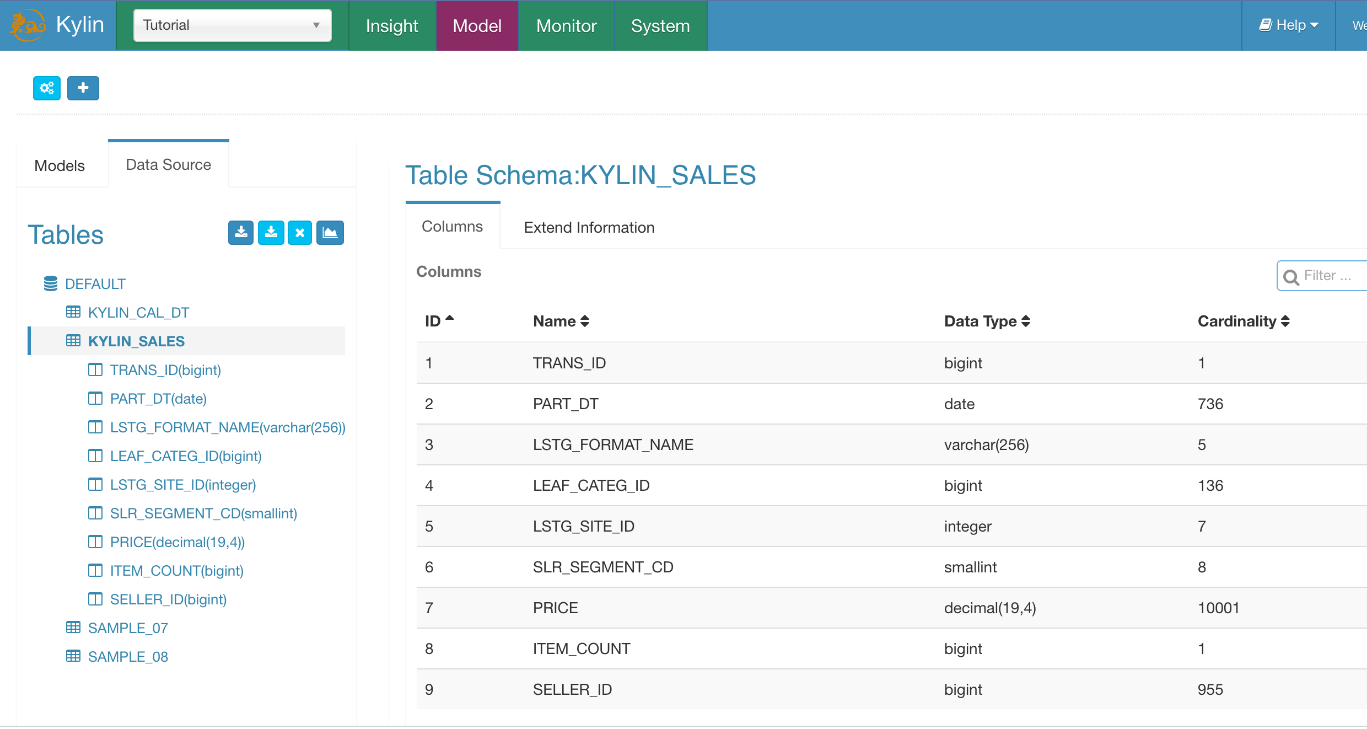
### 访问 & 登录

访问主机: http://hostname:7070  
使用用户名登陆：ADMIN  
使用密码登陆：KYLIN



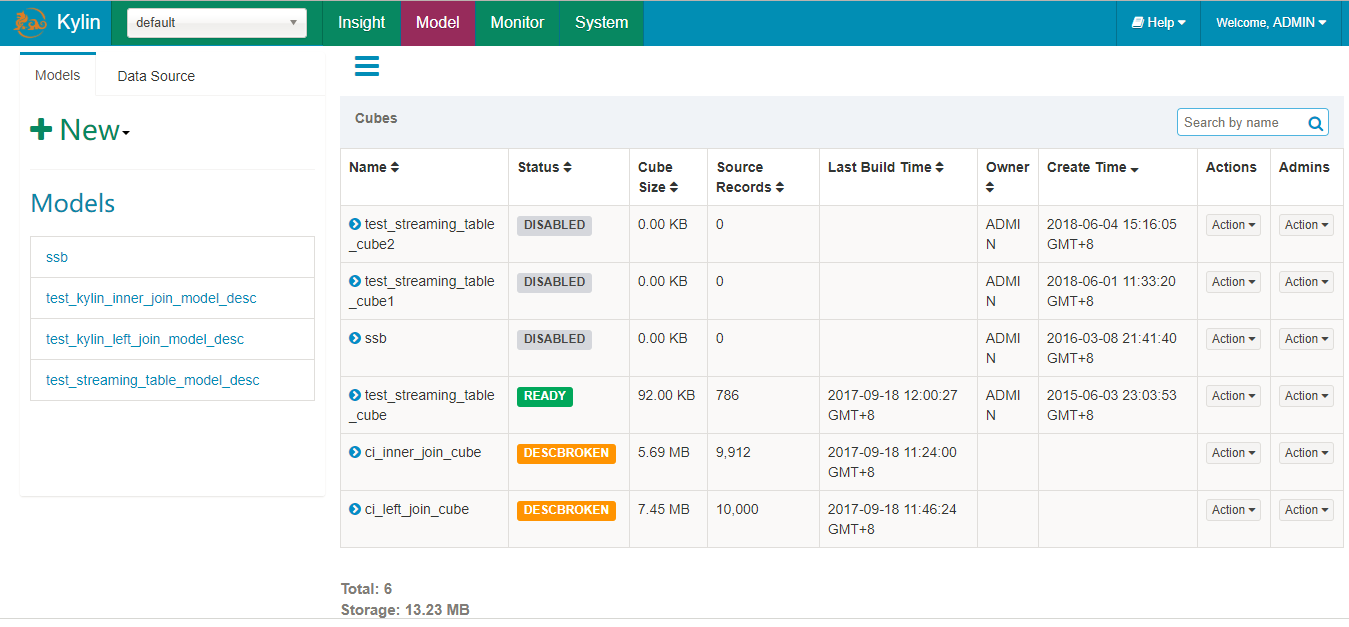
### Kylin中可用的Hive表

虽然 Kylin 使用 SQL 作为查询接口并利用 Hive 元数据，Kylin 不会让用户查询所有的 hive 表，因为到目前为止它是一个预构建 OLAP(MOLAP) 系统。为了使表在 Kylin 中可用，使用 “Sync” 方法能够方便地从 Hive 中同步表。



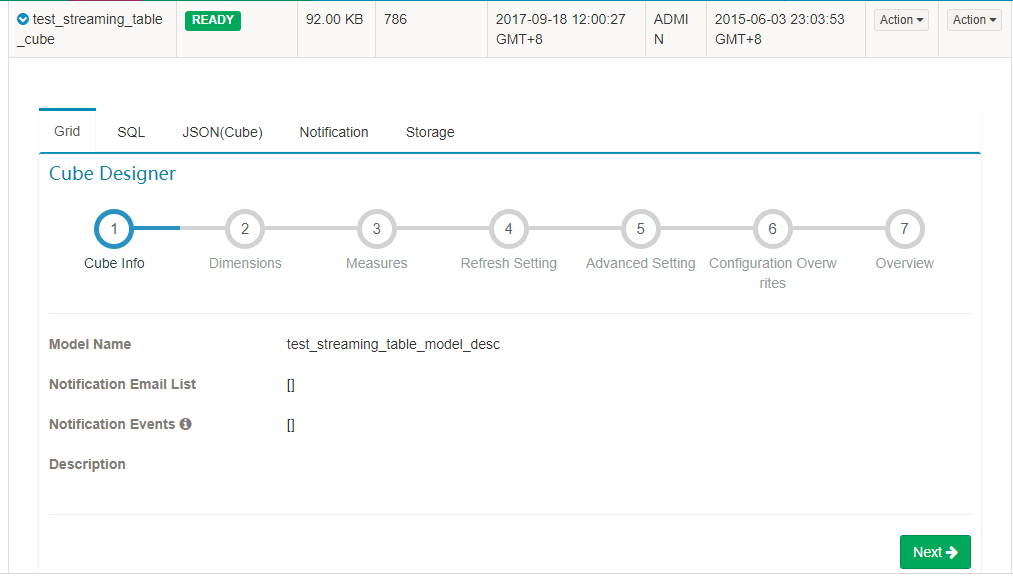
### Kylin OLAP Cube

Kylin 的 OLAP Cube 是从星型模式的 Hive 表中获取的预计算数据集，这是供用户探索、管理所有 cube 的网页管理页面。由菜单栏进入 Model 页面，系统中所有可用的 cube 将被列出。

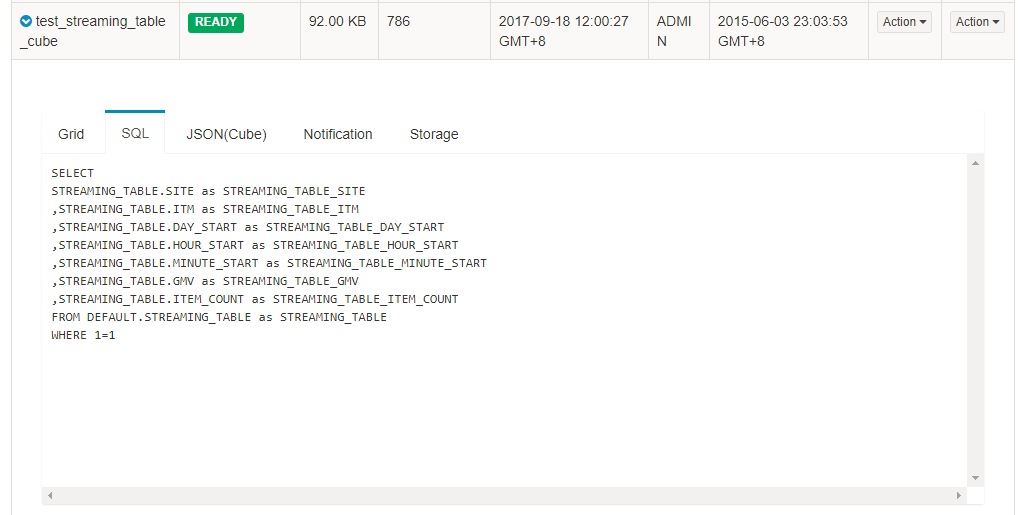


探索更多关于Cube的详细信息

Grid视图



SQL视图（Hive查询读取数据以生成cube）



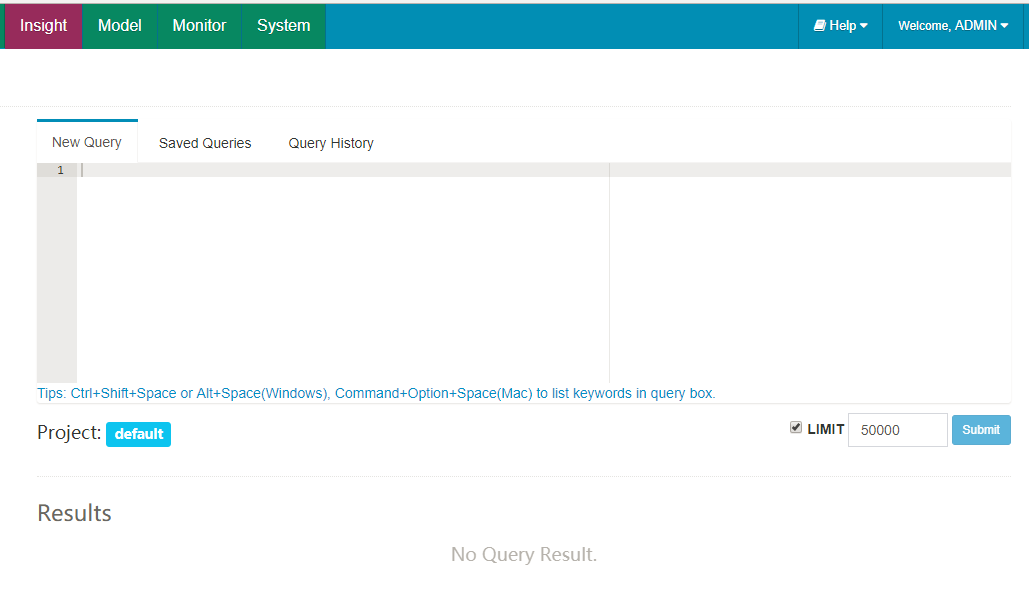
### 在网页上编写和运行SQL

Kylin 的网页版为用户提供了一个简单的查询工具来运行 SQL 以探索现存的 cube，验证结果并探索使用下一章中的 Pivot analysis 与可视化的结果集。

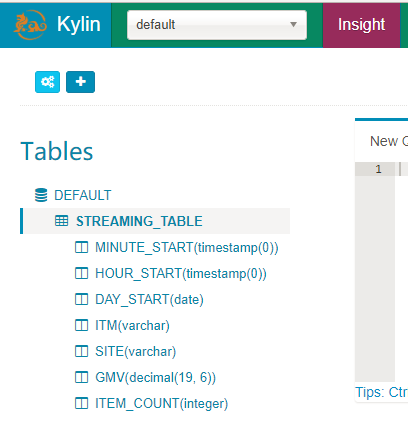
**查询限制**

1. 仅支持 SELECT 查询
2. 支持聚合函数和 group by

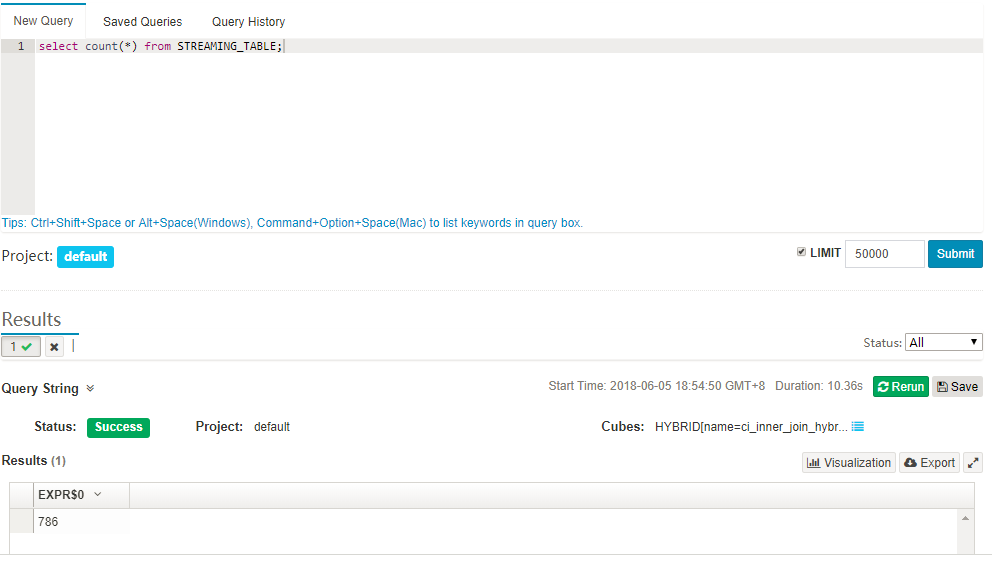
由菜单栏进入 “Insight” 页面：



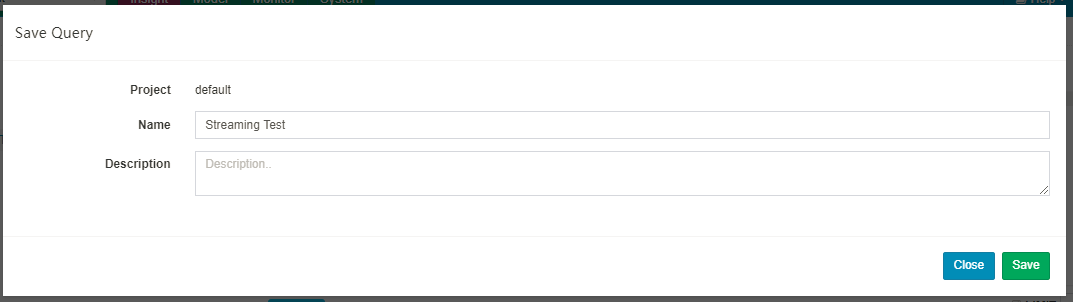
源表：浏览器当前可用表（与Hive相同的结构和元数据）



新的查询：编写和运行查询并探索结果



已保存的查询：与用户账号关联，你将能够从不同的浏览器甚至机器上获取已保存的查询。在结果区域点击 “Save”，将会弹出用来保存当前查询名字和描述。点击 “Saved Queries” 浏览所有已保存的查询，你可以直接重新提交它或删除它。



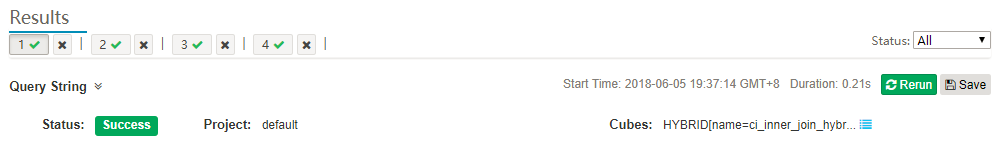
查询历史：仅保存当前用户在当前浏览器中的查询历史，这将需要启用 cookie，并且如果你清理浏览器缓存将会丢失数据。点击 “Query History” 标签，你可以直接重新提交其中的任何一条并再次运行。

### Pivot Analysis与可视化

Kylin 的网页版提供一个简单的 Pivot 与可视化分析工具供用户探索他们的查询结果：

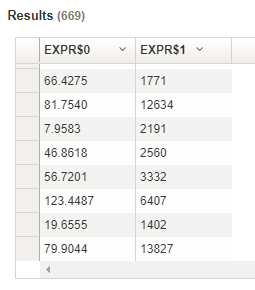
* 一般信息：

当查询运行成功后，它将呈现一个成功指标与被访问的 cube 名字。同时它将会呈现这个查询在后台引擎运行了多久（不包括从 Kylin 服务器到浏览器的网络通信）：



* 查询结果：

能够方便地在一个列上排序。



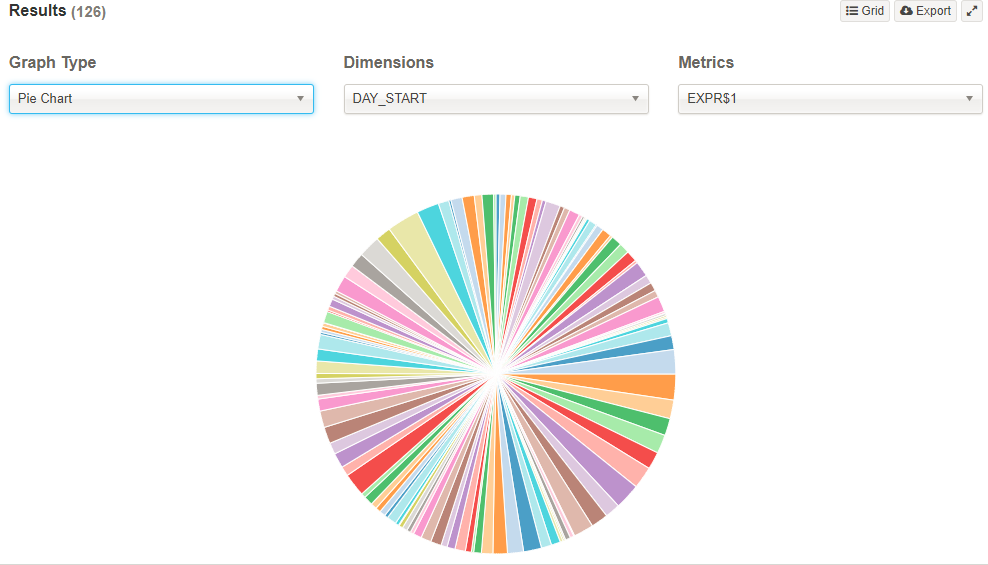
* 导出到 CSV 文件

点击 “Export” 按钮以 CSV 文件格式保存当前结果。

* 可视化：

同时，结果集将被方便地显示在 “可视化” 的不同图表中，总共有3种类型的图表：线性图、饼图和条形图。注意：线形图仅当至少一个从 Hive 表中获取的维度有真实的 “Date” 数据类型列时才是可用的。

* 饼图：



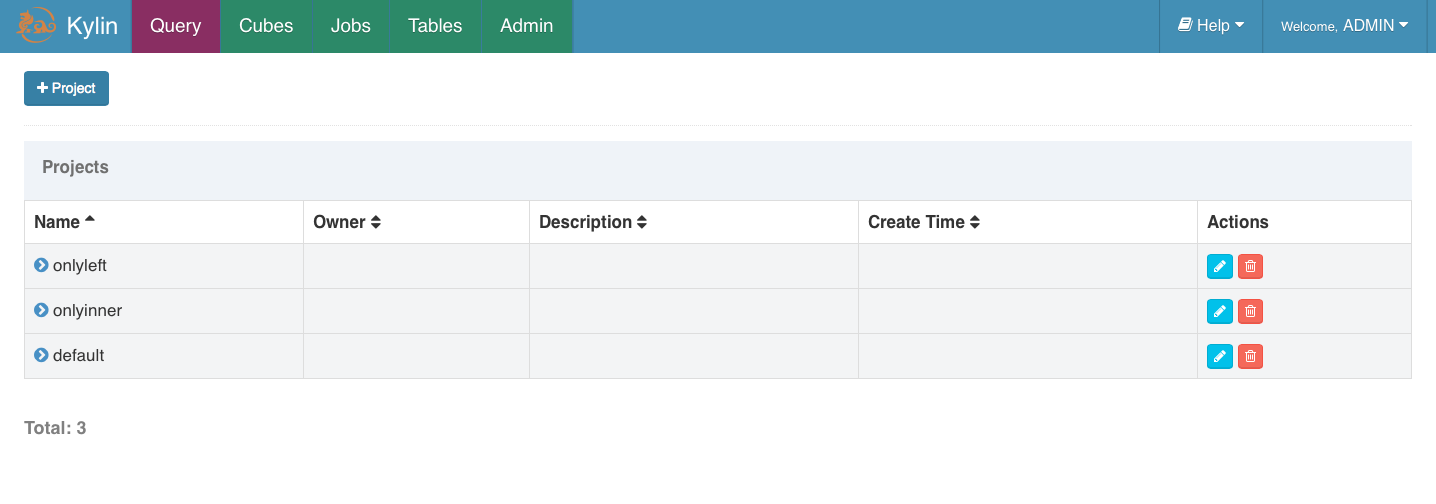
## Cube创建

### 新建项目

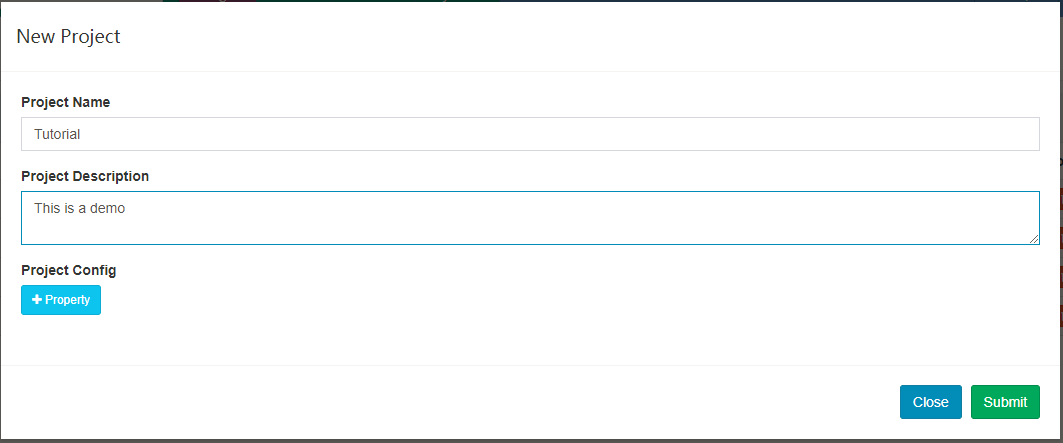
1. 由顶部菜单栏进入 Model 页面，然后点击 Manage Projects。

http://kylin.apache.org/images/tutorial/1.5/Kylin-Cube-Creation-Tutorial/1%20manage-prject.png

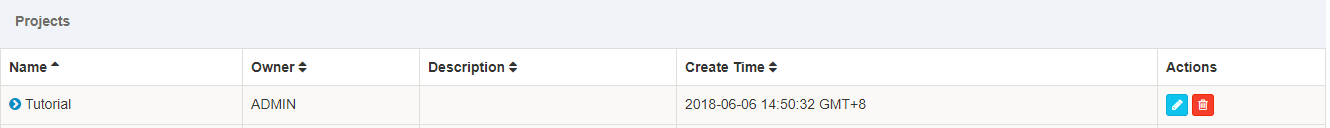
1. 点击 + Project 按钮添加一个新的项目。



1. 填写下列表单并点击submit按钮提交请求



1. 成功后，底部会显示通知

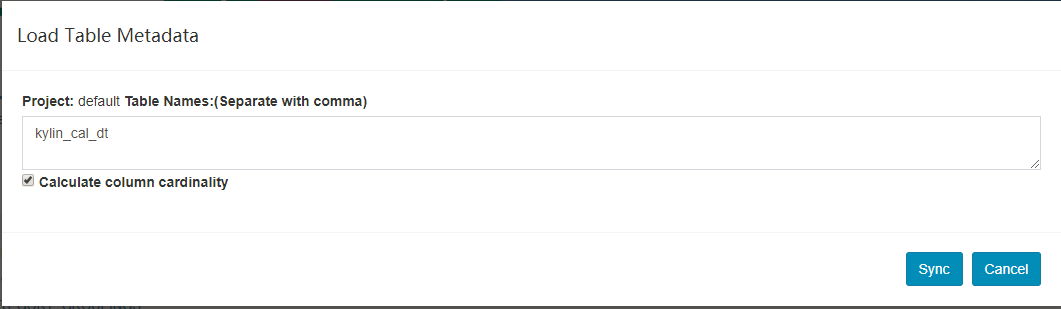


### 同步Hive表

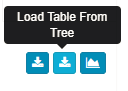
1. 在顶部菜单栏点击 Model，然后点击左边的 Data Source 标签，它会列出所有加载进 Kylin 的表，点击 Load Table 按钮。



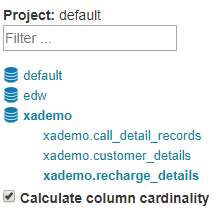
1. 输入表名并点击 Sync 按钮提交请求。



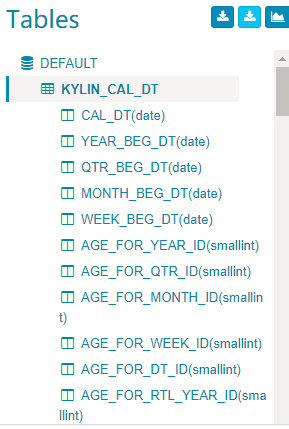
1. 【可选】如果你想要浏览 hive 数据库来选择表，点击 Load Table From Tree 按钮。



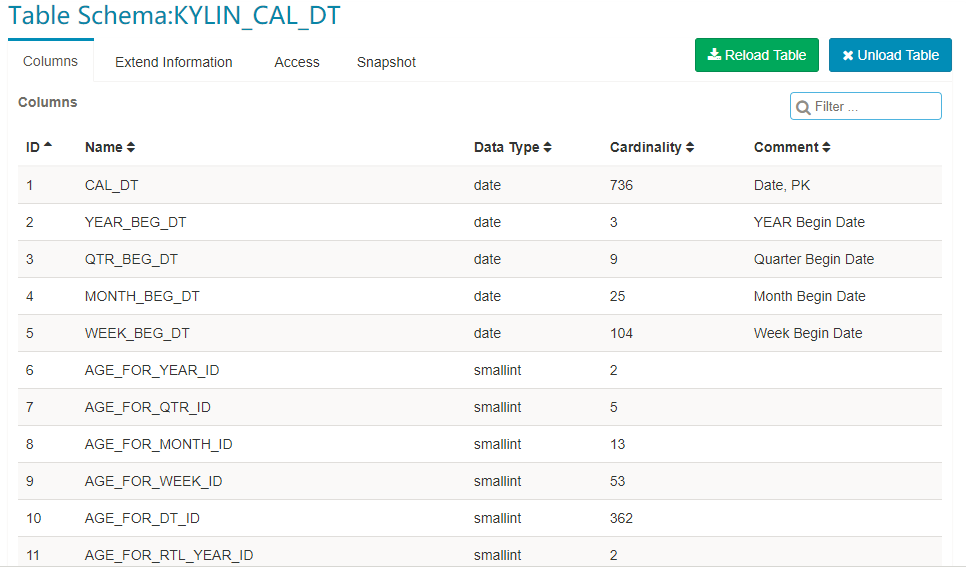
1. 【可选】展开数据库节点，点击选择要加载的表，然后点击 Sync 按钮。



1. 成功的消息将会弹出，在左边的 Tables 部分，新加载的表已经被添加进来。点击表将会展开列。



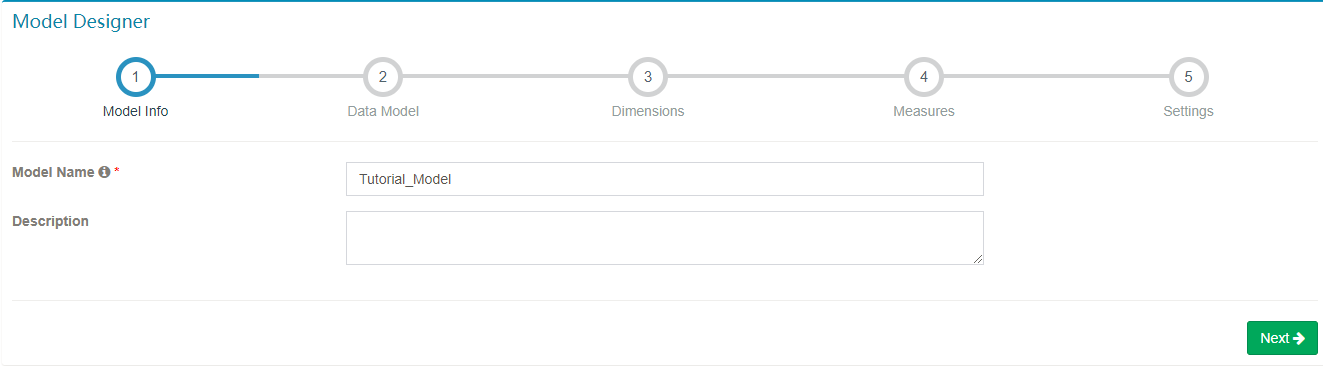
1. 在后台，Kylin 将会执行 MapReduce 任务计算新同步表的基数（cardinality），任务完成后，刷新页面并点击表名，基数值将会显示在表信息中。



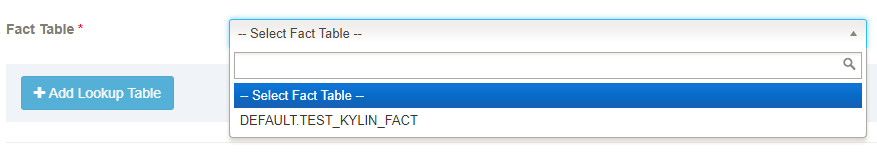
### 新建Data Model

创建 cube 前，需定义一个数据模型。数据模型定义了一个星型（star schema）或雪花（snowflake schema）模型。一个模型可以被多个 cube 使用。

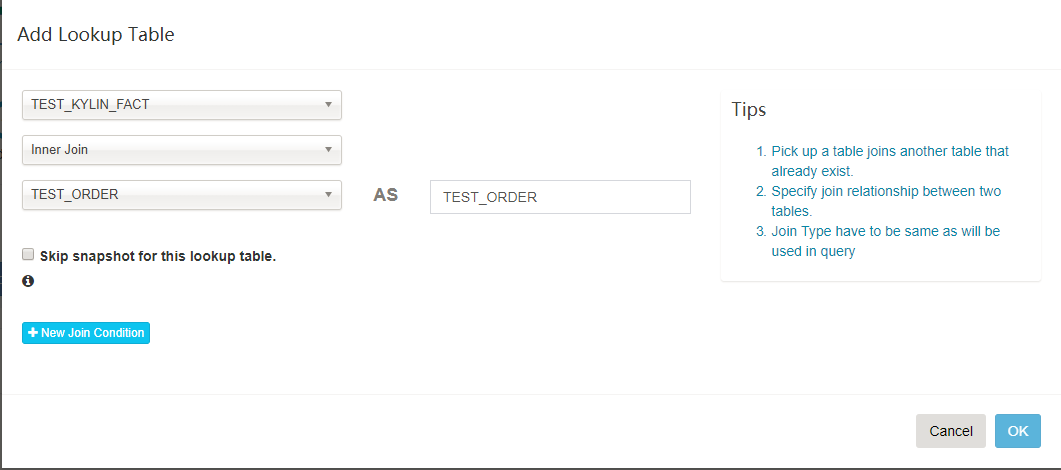
1. 点击顶部的 Model ，然后点击 Models 标签。点击 +New 按钮，在下拉框中选择 New Model。
2. 输入 model 的名字和可选的描述。



1. 在 Fact Table 中，为模型选择事实表。



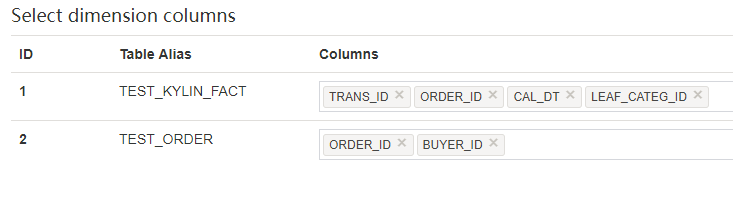
1. 【可选】点击 Add Lookup Table 按钮添加一个 lookup 表。选择表名和关联类型（内连接或左连接）



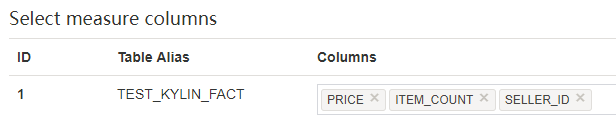
1. 点击 New Join Condition 按钮，左边选择事实表的外键，右边选择 lookup 表的主键。如果有多于一个 join 列重复执行。



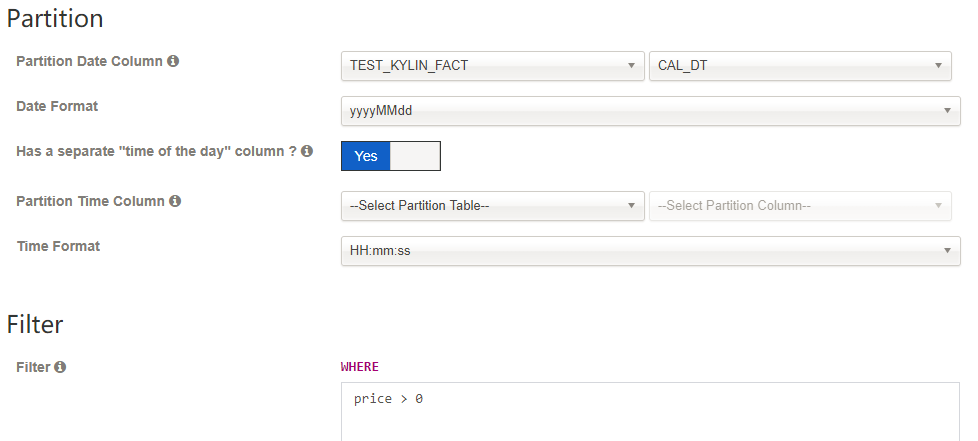
1. 点击 “OK”，重复4，5步来添加更多的 lookup 表。完成后，点击 “Next”。
2. Dimensions 页面允许选择在子 cube 中用作维度的列，然后点击 Columns 列，在下拉框中选择需要的列。



1. 点击 “Next” 到达 “Measures” 页面，选择作为 measure 的列，其只能从事实表中选择。



1. 点击 “Next” 到达 “Settings” 页面，如果事实表中的数据每日增长，选择 Partition Date Column 中相应的 日期列以及日期格式，否则就将其留白。
2. 【可选】选择是否需要 “time of the day” 列，默认情况下为 No。如果选择 Yes, 选择 Partition Time Column 中相应的 time 列以及 time 格式



1. 【可选】如果在从 hive 抽取数据时候想做一些筛选，可以在 Filter 中输入筛选条件。
2. 点击 Save 然后选择 Yes 来保存 data model。创建完成，data model 就会列在左边 Models 列表中。

### 新建Cube

创建完 data model，可以开始创建 cube。点击顶部 Model，然后点击 Models 标签。点击 +New 按钮，在下拉框中选择 New Cube。

**步骤1. Cube 信息**

选择 data model，输入 cube 名字；点击 Next 进行下一步。

cube 名字可以使用字母，数字和下划线（空格不允许）。Notification Email List 是运用来通知job执行成功或失败情况的邮箱列表。Notification Events 是触发事件的状态。

**步骤2. 维度**

1. 点击 Add Dimension，在弹窗中显示的事实表和 lookup 表里勾选输入需要的列。Lookup 表的列有2个选项：“Normal” 和 “Derived”（默认）。“Normal” 添加一个普通独立的维度列，“Derived” 添加一个 derived 维度，derived 维度不会计算入 cube，将由事实表的外键推算出。阅读更多【如何优化 cube】（/docs15/howto/howto\_optimize\_cubes.html）。
2. 选择所有维度后点击 “Next”。

**步骤3. 度量**

1. 点击 +Measure 按钮添加一个新的度量。
2. 根据它的表达式共有8种不同类型的度量：SUM、MAX、MIN、COUNT、COUNT\_DISTINCT TOP\_N, EXTENDED\_COLUMN 和 PERCENTILE。请合理选择 COUNT\_DISTINCT 和 TOP\_N 返回类型，它与 cube 的大小相关。

**步骤4. 更新设置**

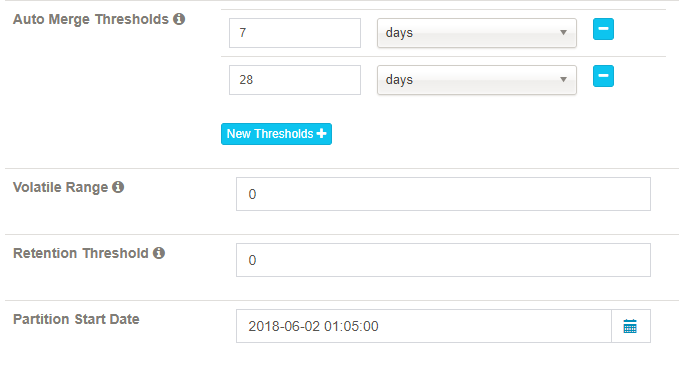
这一步骤是为增量构建 cube 而设计的。

Auto Merge Thresholds: 自动合并小的 segments 到中等甚至更大的 segment。如果不想自动合并，删除默认2个选项。

Volatile Range: 默认为0，会自动合并所有可能的 cube segments，或者用 ‘Auto Merge’ 将不会合并最新的 [Volatile Range] 天的 cube segments。

Retention Threshold: 只会保存 cube 过去几天的 segment，旧的 segment 将会自动从头部删除；0表示不启用这个功能。

Partition Start Date: cube 的开始日期.



**步骤5. 高级设置**

Aggregation Groups: Cube 中的维度可以划分到多个聚合组中。默认 kylin 会把所有维度放在一个聚合组，当维度较多时，产生的组合数可能是巨大的，会造成 Cube 爆炸；如果你很好的了解你的查询模式，那么你可以创建多个聚合组。在每个聚合组内，使用 “Mandatory Dimensions”, “Hierarchy Dimensions” 和 “Joint Dimensions” 来进一步优化维度组合。

Mandatory Dimensions: 必要维度，用于总是出现的维度。例如，如果你的查询中总是会带有 “ORDER\_DATE” 做为 group by 或 过滤条件, 那么它可以被声明为必要维度。这样一来，所有不含此维度的 cuboid 就可以被跳过计算。

Hierarchy Dimensions: 层级维度，例如 “国家” -> “省” -> “市” 是一个层级；不符合此层级关系的 cuboid 可以被跳过计算，例如 [“省”], [“市”]. 定义层级维度时，将父级别维度放在子维度的左边。

Joint Dimensions:联合维度，有些维度往往一起出现，或者它们的基数非常接近（有1:1映射关系）。例如 “user\_id” 和 “email”。把多个维度定义为组合关系后，所有不符合此关系的 cuboids 会被跳过计算。

关于更多维度优化，请阅读这个博客: [新的聚合组](http://kylin.apache.org/blog/2016/02/18/new-aggregation-group/)

Rowkeys: 是由维度编码值组成。”Dictionary” （字典）是默认的编码方式; 字典只能处理中低基数（少于一千万）的维度；如果维度基数很高（如大于1千万), 选择 “false” 然后为维度输入合适的长度，通常是那列的最大长度值; 如果超过最大值，会被截断。请注意，如果没有字典编码，cube 的大小可能会非常大。

你可以拖拽维度列去调整其在 rowkey 中位置; 位于rowkey前面的列，将可以用来大幅缩小查询的范围。通常建议将 mandantory 维度放在开头, 然后是在过滤 ( where 条件)中起到很大作用的维度；如果多个列都会被用于过滤，将高基数的维度（如 user\_id）放在低基数的维度（如 age）的前面。

Mandatory Cuboids: 维度组合白名单。确保你想要构建的 cuboid 能被构建。

Cube Engine: cube 构建引擎。有两种：MapReduce 和 Spark。如果你的 cube 只有简单度量（SUM, MIN, MAX)，建议使用 Spark。如果 cube 中有复杂类型度量（COUNT DISTINCT, TOP\_N），建议使用 MapReduce。

Advanced Dictionaries: “Global Dictionary” 是用于精确计算 COUNT DISTINCT 的字典, 它会将一个非 integer的值转成 integer，以便于 bitmap 进行去重。如果你要计算 COUNT DISTINCT 的列本身已经是 integer 类型，那么不需要定义 Global Dictionary。 Global Dictionary 会被所有 segment 共享，因此支持在跨 segments 之间做上卷去重操作。请注意，Global Dictionary 随着数据的加载，可能会不断变大。

“Segment Dictionary” 是另一个用于精确计算 COUNT DISTINCT 的字典，与 Global Dictionary 不同的是，它是基于一个 segment 的值构建的，因此不支持跨 segments 的汇总计算。如果你的 cube 不是分区的或者能保证你的所有 SQL 按照 partition\_column 进行 group by, 那么你应该使用 “Segment Dictionary” 而不是 “Global Dictionary”，这样可以避免单个字典过大的问题。

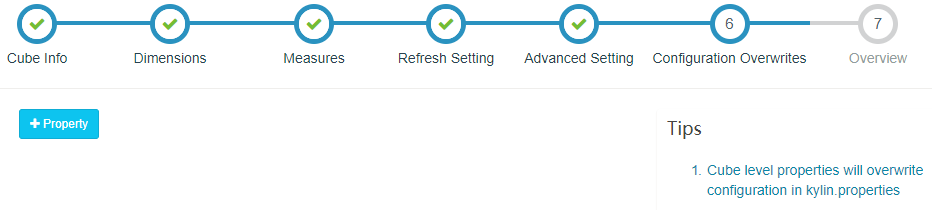
请注意：”Global Dictionary” 和 “Segment Dictionary” 都是单向编码的字典，仅用于 COUNT DISTINCT 计算(将非 integer 类型转成 integer 用于 bitmap计算)，他们不支持解码，因此不能为普通维度编码。

Advanced Snapshot Table: 为全局 lookup 表而设计，提供不同的存储类型。

Advanced ColumnFamily: 如果有超过一个的COUNT DISTINCT 或 TopN 度量, 你可以将它们放在更多列簇中，以优化与HBase 的I/O。

**步骤6. 重写配置**

Kylin 允许在 Cube 级别覆盖部分 kylin.properties 中的配置，你可以在这里定义覆盖的属性。如果你没有要配置的，点击 Next 按钮。



**步骤7. 概览 & 保存**

你可以概览你的 cube 并返回之前的步骤进行修改。点击 Save 按钮完成 cube 创建。

## Cube创建和Job监控

## SQL快速参考

Apache Kylin 使用 Apache Calcite 做 SQL 解析和优化。作为一款 OLAP 引擎, Kylin 支持 SELECT 操作，而不支持其它操作例如 INSERT，UPDATE 和 DELETE，因此 Kylin 的 SQL 语法是 Apache Calcite 支持语法的一个子集。本文列举了 Kylin 支持的 SQL 语法、函数以及数据类型，但可能并不完整。您可以查看 [Calcite SQL reference](https://calcite.apache.org/docs/reference.html) 以了解更多内容。

## 项目和表级别权限控制

### 项目级别权限控制

用户是否可以访问一个项目并使用项目中的功能取决于项目级别的权限控制，Kylin 中共有 4 种角色。分别是 ADMIN，MANAGEMENT，OPERATION 和 QUERY。每个角色对应不同的功能。

* QUERY：适用于只需在项目中有查询表/cube 权限的分析师。
* OPERATION：该角色适用于需维护 Cube 的公司/组织中的运营团队。OPERATION 包含 QUERY 的所有权限。
* MANAGEMENT：该角色适用于充分了解数据/模型商业含义的模型师，建模师会负责模型和 Cube 的设计。MANAGEMENT 包含 OPERATION 和 QUERY 的所有权限。
* ADMIN：该角色全权管理项目。ADMIN 包含 MANAGEMENT，OPERATION 和 QUERY 的所有权限。

访问权限是项目隔离的。

### 如何确定访问权限

为用户设置项目级别的访问权限后，不同的角色对应于不同的对数据源，模型和 Cube 的访问权限。具体的功能，以及每个角色的访问权限，如下表所示。

|  | **System Admin** | **Project Admin** | **Management** | **Operation** | **Query** |
| --- | --- | --- | --- | --- | --- |
| Create/delete project | Yes | No | No | No | No |
| Edit project | Yes | Yes | No | No | No |
| Add/edit/delete project access permission | Yes | Yes | No | No | No |
| Check model page | Yes | Yes | Yes | Yes | Yes |
| Check data source page | Yes | Yes | Yes | No | No |
| Load, unload table, reload table | Yes | Yes | No | No | No |
| View model in read only mode | Yes | Yes | Yes | Yes | Yes |
| Add, edit, clone, drop model | Yes | Yes | Yes | No | No |
| Check cube detail definition | Yes | Yes | Yes | Yes | Yes |
| Add, disable/enable, clone cube, edit, drop cube, purge cube | Yes | Yes | Yes | No | No |
| Build, refresh, merge cube | Yes | Yes | Yes | Yes | No |
| Edit, view cube json | Yes | Yes | Yes | No | No |
| Check insight page | Yes | Yes | Yes | Yes | Yes |
| View table in insight page | Yes | Yes | Yes | Yes | Yes |
| Check monitor page | Yes | Yes | Yes | Yes | No |
| Check system page | Yes | No | No | No | No |
| Reload metadata, disable cache, set config, diagnosis | Yes | No | No | No | No |

另外，当查询下压开启时，该项目的查询权限允许用户查询项目中的所有表即使没有 cube 为他服务。每个用户都会被授予查询权限。

### 管理项目级别的访问权限

### 管理表级别权限

## 用Spark构建Cube

## 用Kafka流构建Cube

## 优化Cube构建

## Python客户端

Apache Kylin Python 客户端工具库是基于Python可访问Kylin的客户端. 此工具库包含两个可使用组件.

* Apache Kylin 命令行工具
* Apache Kylin SQLAchemy方言

想要了解更多关于此工具库信息请点击[Github仓库](https://github.com/Kyligence/kylinpy).

### 安装

请确保您python解释器版本在2.7+, 或者3.4+以上. 最方便安装Apache Kylin Python客户端工具库的方法是使用pip命令  
pip install --upgrade kylinpy

### Kylinpy 命令行工具

安装完kylinpy后, 立即可以在终端下访问kylinpy

$ kylinpy

Usage: kylinpy [OPTIONS] COMMAND [ARGS]...

Options:

-h, --host TEXT Kylin host name [required]

-P, --port INTEGER Kylin port, default: 7070

-u, --username TEXT Kylin username [required]

-p, --password TEXT Kylin password [required]

--project TEXT Kylin project [required]

--prefix TEXT Kylin RESTful prefix of url, default: /kylin/api

--debug / --no-debug show debug infomation

--api1 / --api2 API version; default is "api1"; "api1" 适用于 Apache Kylin

--help Show this message and exit.

Commands:

auth get user auth info

cube\_columns list cube columns

cube\_desc show cube description

cube\_names list cube names

model\_desc show model description

projects list all projects

query sql query

table\_columns list table columns

table\_names list all table names

### Kylinpy命令行工具示例

1. 访问Apache Kylin  
   kylinpy -h hostname -P 7070 -u ADMIN -p KYLIN --project learn\_kylin --api1 --debug auth
2. 访问选定cube所有的维度信息  
   kylinpy -h hostname -P 7070 -u ADMIN -p KYLIN --project learn\_kylin --api1 --debug cube\_columns --name kylin\_sales\_cube
3. 访问选定的cube描述  
   kylinpy -h hostname -P 7070 -u ADMIN -p KYLIN --project learn\_kylin --api1 --debug cube\_desc --name kylin\_sales\_cube
4. 访问所有cube名称  
   kylinpy -h hostname -u ADMIN -p KYLIN --project learn\_kylin --api1 --debug cube\_names
5. 访问选定cube的SQL定义  
   kylinpy -h hostname -P 7070 -u ADMIN -p KYLIN --project learn\_kylin --api1 --debug cube\_sql --name kylin\_sales\_cube
6. 列出Kylin中所有项目  
   kylinpy -h hostname -P 7070 -u ADMIN -p KYLIN --project learn\_kylin --api1 --debug projects
7. 访问选定表所有的维度信息  
   kylinpy -h hostname -P 7070 -u ADMIN -p KYLIN --project learn\_kylin --api1 --debug table\_columns --name KYLIN\_SALES
8. 访问所有表名  
   kylinpy -h hostname -u ADMIN -p KYLIN --project learn\_kylin --api1 table\_names
9. 访问所选模型信息  
   kylinpy -h hostname -P 7070 -u ADMIN -p KYLIN --project learn\_kylin --api1 --debug model\_desc --name kylin\_sales\_model

### Apache Kylin SQLAlchemy方言

任何一个使用SQLAlchemy的应用程序都可以通过此方言访问到Kylin, 您之前如果已经安装了kylinpy那么现在就已经集成好了SQLAlchemy Dialect. 请使用如下DSN模板访问Kylin

kylin://<username>:<password>@<hostname>:<port>/<project>?version=<v1|v2>&prefix=</kylin/api>

### SQLAlchemy 实例

测试Apache Kylin连接

$ python

>>> import sqlalchemy as sa

>>> kylin\_engine = sa.create\_engine('kylin://username:password@hostname:7070/learn\_kylin?version=v1')

>>> results = kylin\_engine.execute('SELECT count(\*) FROM KYLIN\_SALES')

>>> [e for e in results]

[(4953,)]

>>> kylin\_engine.table\_names()

[u'KYLIN\_ACCOUNT',

u'KYLIN\_CAL\_DT',

u'KYLIN\_CATEGORY\_GROUPINGS',

u'KYLIN\_COUNTRY',

u'KYLIN\_SALES',

u'KYLIN\_STREAMING\_TABLE']

### 使用 Python 和 Apache Kylin 做数据科学分析

请参考此博客：[Use Python for Data Science with Apache Kylin](http://kylin.apache.org/blog/2019/06/26/use-python-for-data-science-with-apache-kylin/).

## 建立系统Cube

### 什么是系统 Cube

为了更好的支持自我监控，在系统 project 下创建一组系统 Cubes，叫做 “KYLIN\_SYSTEM”。现在，这里有五个 Cubes。三个用于查询指标，”METRICS\_QUERY”， ”METRICS\_QUERY\_CUBE”， ”METRICS\_QUERY\_RPC”。另外两个是 job 指标，”METRICS\_JOB”，”METRICS\_JOB\_EXCEPTION”。

## 使用Cube Planner

### 什么是 Cube Planner

OLAP 解决方案权衡了线上查询速度和线下 Cube build 花费（build Cube 的计算资源及保存 Cube 数据的存储资源）。资源效率是 OLAP engine 的最重要的能力。为了提高资源利用率，pre-build 最有价值的 cuboids 是至关重要的。

Cube Planner 使 Apache Kylin 变得更节约资源。其智能 build 部分 Cube 以最小化 building Cube 的花费且同时最大化服务终端用户查询的利益，然后从运行中的查询学习模式且相应的进行动态的推荐 cuboids。

## 使用Dashboard

## 建立JDBC数据源

## Hybrid模型

## 基于MySQL的Metastore配置

# 工具集成

## ODBC驱动

我们提供Kylin ODBC驱动程序以支持ODBC兼容客户端应用的数据访问。

32位版本或64位版本的驱动程序都是可用的。

测试操作系统：Windows 7，Windows Server 2008 R2

测试应用：Tableau 8.0.4 和 Tableau 8.1.3

### 前提条件

1. Microsoft Visual C++ 2012 再分配（Redistributable）

32位Windows或32位Tableau Desktop：下载：[32bit version](http://download.microsoft.com/download/1/6/B/16B06F60-3B20-4FF2-B699-5E9B7962F9AE/VSU_4/vcredist_x86.exe)

64位Windows或64位Tableau Desktop：下载：[64bit version](http://download.microsoft.com/download/1/6/B/16B06F60-3B20-4FF2-B699-5E9B7962F9AE/VSU_4/vcredist_x64.exe)

1. ODBC驱动程序内部从一个REST服务器获取结果，确保你能够访问一个

### 安装

1. 如果你已经安装，首先卸载已存在的Kylin ODBC
2. 从[下载](http://kylin.apache.org/cn/download/)下载附件驱动安装程序，并运行。

32位Tableau Desktop：请安装KylinODBCDriver (x86).exe

64位Tableau Desktop：请安装KylinODBCDriver (x64).exe

1. Both drivers already be installed on Tableau Server, you properly should be able to publish to there without issues

## JDBC驱动

### 认证

基于Apache Kylin认证RESTFUL服务。支持的参数：

* user : 用户名
* password : 密码
* ssl: true或false。 默认为flas；如果为true，所有的服务调用都会使用https。

### 连接url格式：

jdbc:kylin://<hostname>:<port>/<kylin\_project\_name>

* 如果“ssl”为true，“port”应该是Kylin server的HTTPS端口。
* 如果“port”未被指定，driver会使用默认的端口：HTTP 80，HTTPS 443。
* 必须指定“kylin\_project\_name”并且用户需要确保它在Kylin server上存在。

### 使用Statement查询

Driver driver = (Driver) Class.forName("org.apache.kylin.jdbc.Driver").newInstance();

Properties info = new Properties();

info.put("user", "ADMIN");

info.put("password", "KYLIN");

Connection conn = driver.connect("jdbc:kylin://localhost:7070/kylin\_project\_name", info);

Statement state = conn.createStatement();

ResultSet resultSet = state.executeQuery("select \* from test\_table");

while (resultSet.next()) {

assertEquals("foo", resultSet.getString(1));

assertEquals("bar", resultSet.getString(2));

assertEquals("tool", resultSet.getString(3));

}

### 使用PreparedStatementv查询

支持的PreparedStatement参数：

* setString
* setInt
* setShort
* setLong
* setFloat
* setDouble
* setBoolean
* setByte
* setDate
* setTime
* setTimestamp

Driver driver = (Driver) Class.forName("org.apache.kylin.jdbc.Driver").newInstance();

Properties info = new Properties();

info.put("user", "ADMIN");

info.put("password", "KYLIN");

Connection conn = driver.connect("jdbc:kylin://localhost:7070/kylin\_project\_name", info);

PreparedStatement state = conn.prepareStatement("select \* from test\_table where id=?");

state.setInt(1, 10);

ResultSet resultSet = state.executeQuery();

while (resultSet.next()) {

assertEquals("foo", resultSet.getString(1));

assertEquals("bar", resultSet.getString(2));

assertEquals("tool", resultSet.getString(3));

}

### 获取查询结果元数据

Kylin jdbc driver支持元数据列表方法：通过sql模式过滤器（比如 %）列出catalog、schema、table和column。

Driver driver = (Driver) Class.forName("org.apache.kylin.jdbc.Driver").newInstance();

Properties info = new Properties();

info.put("user", "ADMIN");

info.put("password", "KYLIN");

Connection conn = driver.connect("jdbc:kylin://localhost:7070/kylin\_project\_name", info);

Statement state = conn.createStatement();

ResultSet resultSet = state.executeQuery("select \* from test\_table");

ResultSet tables = conn.getMetaData().getTables(null, null, "dummy", null);

while (tables.next()) {

for (int i = 0; i < 10; i++) {

assertEquals("dummy", tables.getString(i + 1));

}

}

## Tableau8

## Tableau9

## Excel及PowerBI教程

Microsoft Excel是当今Windows平台上最流行的数据处理软件之一，支持多种数据处理功能，可以利用Power Query从ODBC数据源读取数据并返回到数据表中。

Microsoft Power BI 是由微软推出的商业智能的专业分析工具，给用户提供简单且丰富的数据可视化及分析功能。

* Apache Kylin目前版本不支持原始数据的查询，部分查询会因此失败，导致应用程序发生异常，建议打上KYLIN-1075补丁包以优化查询结果的显示。
* Power BI及Excel不支持”connect live”模式，请注意并添加where条件在查询超大数据集时候，以避免从服务器拉去过多的数据到本地，甚至在某些情况下查询执行失败。

## Squirrel

[SQuirreL SQL](http://www.squirrelsql.org/)是一个多平台通用SQL客户端（GNU许可证）。您可以使用它来访问HBase + Phoenix和Hive。本文档介绍了如何从SQuirreL连接到Kylin。

## Qlik Sense集成

Qlik Sense 是新一代自助式数据可视化工具。它是一款完整的商业分析软件，便于开发人员和分析人员快速构建和部署强大的分析应用。近年来，该工具成为全球增长率最快的 BI 产品。它可以与 Hadoop Database（Hive 和 Impala）集成。现在也可与 Apache Kylin 集成。

* 安装Kylin ODBC驱动程序
* 安装Qlik Sense
* 与Qlik Sense连接
* 配置Direct Query连接模式
* 创建报表

## Superset

Apache Kylin 与 Apache Superset 的集成

### 简介

Apache Superset（incubating）是一个现代化的企业级商业智能 Web 应用程序。Superset 的整个后端是基于 Python 开发的，用到了 Flask，Pandas，SqlAlchemy。可以与 Kylin Python Client 集成。

**Apache Superset 功能**

* 丰富的数据可视化集
* 易于使用的界面，用于探索和可视化数据
* 创建和共享仪表板
* 与主要身份验证提供程序（数据库，OpenID，LDAP，通过 Flask AppBuilder 的 OAuth 和 REMOTE\_USER）集成的企业级身份验证
* 一种可扩展的，高粒度的安全/权限模型，对于可以访问个人特征和数据集的用户允许使用复杂的规则
* 一个简单的语义层，允许用户通过定义哪些字段应显示在哪个下拉列表以及哪些聚合和功能度量标准可供用户使用来控制数据源在 UI 中的显示方式
* 通过 SQLAlchemy 与大多数 SQL 的 RDBMS 集成

### 集成的好处

Apache Kylin 和 Apache Superset 都是以为其用户提供更快和可交互式的分析的目的而构建。由于预先计算的 Kylin Cube，在 PB 级数据集上这两个开源项目的结合可以将这个目标变为现实。

### 集成的步骤

1. 安装 Apache Kylin
2. 成功的 build cube
3. 安装 Apache Superset 并进行初始化
4. 在 Apache Superset 中连接 Apache Kylin
5. 配置一个新的数据源
6. 测试及查询

详细的集成步骤，请查看 [这篇文章](http://kylin.apache.org/blog/2018/01/01/kylin-and-superset/)。

### 其它功能

Apache Superset 也支持导出 CSV, 共享, 以及查看 SQL 查询。