Дата последней сборки документа:
27 мая 2021 г. в 22:23

Сборник заметок по аналитической колоночной СУБД с массово-параллельной архитектурой без разделения ресурсов Vertica

Содержание

1. Установка СУБД Vertica

Скачать Community Edition версию `Vertica` можно по ссылке https://www.vertica.com/download/vertica/community-edition/. В разделе Community Edition xx.x.x Virtual Machine следует клинуть на Open Virtualization Format. После чего начнется скачивание OVA-файла Vertica.

OVA-файл это пакет, содержащий файлы, используемые для описания виртуальной машины, который включает файл дескриптора `.OVF`, необязательный файл манифеста `.MF`, файлы сертификатов и другие связанные файлы.

OVA-файлы это файлы Open Virtual Appliance, которые еще иногда называют файлами Open Virtual Application или файлами Open Virtualization Format Archive. Они используются программами виртуализации для хранения различных файлов, связанных с виртуальной машиной.

Другими словами, OVA-файл – это открытое устройство виртуализации, содержащее сжатую версию устанавливаемой виртуальной машины.

Скачав OVA-файл остается только запустить его, например, в VirtualBox. Виртуальная машина `vertica_community_edition-xx.x.x-x.ova` будет содержать собственно Vertica Community Edition, VMart EXample Database, VMart Managment Console, Vertica Administration Tools and vsql и Vertica CE VM User Guide. Начать работать с платформой можно по руководству https://www.vertica.com/docs/VMs/Vertica_CE_VM_User_Guide.pdf.

В целом диалект SQL `Vertica` мало чем отличается от диалекта PostgreSQL, но есть и нюансы, с которыми можно ознакомиться на странице документации https://www.vertica.com/docs/10.0.x/HTML/Content/Home.htm.

Для работы с этой СУБД через помощью Python API есть своя библиотека `vertica_python`[1] https://github.com/vertica/vertica-python

```
import vertica_python

conn_info = {
  'host' : '127.0.0.1',
```

[1] Устанавливается как обычно с помощью менеджера пакетов `pip`: `pip install vertica-python`

```
  'port' : 5433,
  'user' : 'some_user',
  'password' : 'some_password',
  'database' : 'a_database',
  'kerberos_service_name' : 'vertica_krb',
  'kerberos_host_name' : 'vlcuster.example.com'
}

with vertica_python.connect(**conn_info) as conn:
  # do things
```

Вариант с баллансировкой нагрузки

```
import vertica_python

conn_info = {
  'host' : '127.0.0.1',
  'port' : 5433,
  'user' : 'some_user',
  'password' : 'some_password',
  'database' : 'vdb',
  'connection_laod_balance' : True
}

# Server enables load balancing
with vertica_python.connect(**conn_info) as conn:
  cur = conn.cursor()
  cur.execute('SELECT NODE_NAME FROM V_MONITOR.CURRENT_SESSION')
  print('Client connects to primary node:', cur.fetchone()[0])
  cur.execute("SELECT SET_LOAD_BALANCE_POLICY('ROUNDROBIN')")

with vertica_python.connect(**conn_info) as conn:
  cur = conn.cursor()
  cur.execute('SELECT NODE_NAME FROM V_MONITOR.CURRENT_SESSION')
  print('Client redirects to node:', cur.fetchone()[0])
```

А с помощью библиотеки VerticaPy https://github.com/vertica/VerticaPy к хранящимся в СУБД данным можно применять модели машинного обучения. Например

```
import vertica_python
from verticapy.learn.datasets import load_titanic
from verticapy.learn.model_selection import cross_validate
from verticapy.learn.ensemble import RandomForestClassifier

# Connection using all the DSN information
conn_info = {
  'host': "10.211.55.14",
  'port': 5433,
  'user': "dbadmin",
  'password': "XxX",
  'database': "testdb"
}

cur = vertica_python.connect(** conn_info).cursor()

vdf = load_titanic(cursor = cur)

# Data Preparation
vdf["sex"].label_encode()["boat"].fillna(method = "0ifnull")["name"].str_extract(' ([A-Za-z]+)\.
    ').eval("family_size", expr = "parch + sibsp + 1").drop(columns = ["cabin", "body", "ticket"
    , "home.dest"])["fare"].fill_outliers().fillna()
```

```
# Model Evaluation
cross_validate(RandomForestClassifier("rf_titanic", cur, max_leaf_nodes = 100, n_estimators =
    30),
vdf, ["age", "family_size", "sex", "pclass", "fare", "boat"],
"survived",
cutoff = 0.35)
```

2. Вставка нескольких строк в таблицу

`Vertica` не поддерживает вставку нескольких строк в одной инструкции `INSERT INTO` с помощью виртуальных таблиц `VALUES` как, например, PostgreSQL. Тем неменее существует способ вставить в таблицу несколько строк за раз. Сделать это можно с помощью ключевого слова `UNION`

```
INSERT INTO packages(package_name, solver_type)
SELECT 'Ansys', 'direct'
UNION
SELECT 'Nastran', 'iterative'
UNION
SELECT 'Abaqus', 'direct'
```

Список литературы

1. *Джуба С.*, *Волков А.* Изучаем PostgreSQL 10. – М.: ДМК Пресс, 2019. – 400 с.

Листинги