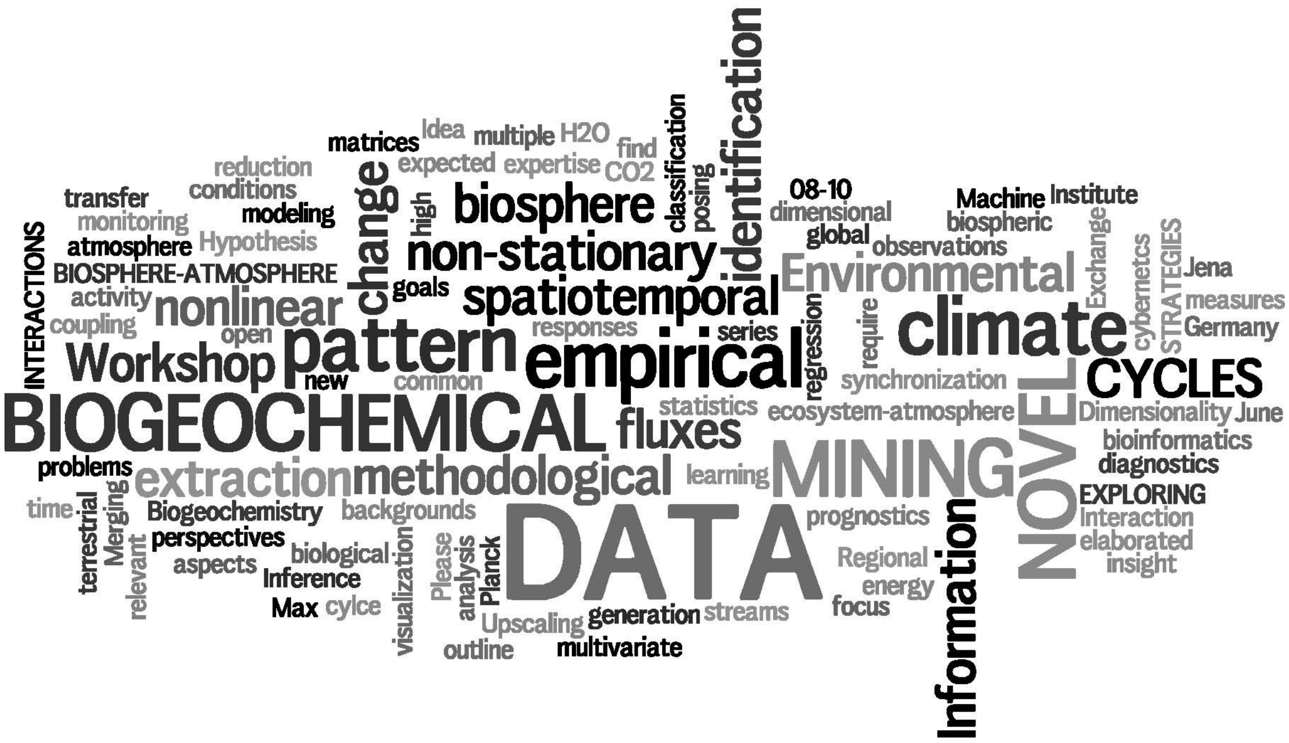
CA4002 CA PROJECT

Web Scraping/Data Warehousing/Data Mining

David O'Regan - 31 March 2014



Introduction

I delcare that all the work contained in this document and project is my own unless otherwise stated and referenced.

My project was defined as scraping data from www.carzone.ie, cleanly inserting my data into a relational database and finally, applying a suitable data mining algorithm to determine something of interest from the gathered data.

Web Scraping

To start my project, my first challenge was to install Scrapy, and then build a working web scraper.

Getting Scrapy

I started by downloading scrappy to my personal Machine. Given I use a MacBook Pro, installing scrapy turned out to be something of an uphill battle. To begin with, the newsiest version of scrapy does not work with the version of python I was currently working with(2.7). To fix this, I was left with a choice of building a older version of scrapy or upgrading my python Library. I chose to download an older build of scrapy instead of trying to upgrade my python Library as after some research I came to learn the newest version of python(3.3) is a unstable install against my OS.

Installing Scrapy

http://doc.scrapy.org/en/latest/intro/install.html

I used pip to install scrapy and then built the missing dependancies I had on my personal machine. Lxml caused major issues when trying to build scrapy as it wouldn't run without the library, though when I went to install the Lxml package, I was met with this error message:

```
In file included from src/lxml/lxml.etree.c:314:
/private/tmp/pip_build_root/lxml/src/lxml/includes/
etree_defs.h:9:10: fatal error: 'libxml/xmlversion.h' file not
found
#include "libxml/xmlversion.h"
         ^
1 error generated.
error: command 'cc' failed with exit status 1
```

To fix this, I needed to do some digging where I came across a rather hacky but workable solution:

```
STATIC_DEPS=true pip install lxml
```

First Spider

My first spider was a basic crawl spider which I have included in my code files. It was a base spider implemented in scrapy that would parse the first page of carzone.ie, and scrape data out of the presented header.

Getting Header Data

To begin with, my spider was restricted to crawling only the header of the page I was currently on, as I lacked the knowledge to enter a link on a page and pull data from that link, then return to the main page and keep scraping more links.

Upgrading To Crawl Spider

My upgrade consisted of first getting my spider to crawl through multiple pages of the carzone,ie website. This was done using a crawl spider, and the Rule implementation of scrapy. Example - Scrape page 1, then continue to page 2.

Requesting More Data

Finally, I spent a long time upgrading my spider to be able to enter specific car adds, return a much richer data supply and then keep scraping through all possible pages until I had a good selection of data to work with i.e. Multiple data pieces(NCT/Mileage/Transmission/Owners/EngineSize/EngineType) for all models of selected car.

This was the final piece of the puzzle when it came to my scraper as I can now base my mining off the really important aspects of what determines a car's price in a market.

How Often To Scrape?

In the end, I moved my MYSQL DB off my laptop and onto my raspberry Pi at home, this let me run my scraping scripts at a very fair rate as to not be banned from carzone.ie.

My Scraper would run 15 pages every hour and when it had exhausted a certain car model, it would switch to the next model in the list I had designed.

MySQL Data Warehousing

Pushing Data Into MYSQL

To begin with, I needed to push my data from my spider into an SQL database. This required four steps,

1) Download and install MYSQL on my local machine
2) Design DB & Tables
3) Install MYSQLdb
4) Design scrapy pipeline to push data into MYSQL

I downloaded and installed MYSQL to my local laptop but after much debate pushed it off to my raspberry pi at home for ease.

I designed my DB and table structure(See DB Optimisation)

After much trouble, I finally was able to install MySQLdb. The main issue being scrapy wouldn't see the import even though my python library contained it. After some digging my solution was thus

It can also crop up if your MySQL client is newer than your MySQL-python package. In my case, I had a libmysqlclient_r.18.dylib on my machine, but not a libmysqlclient_r.16.dylib. Running `pip search mysql` revealed

MySQL-python - Python interface to MySQL INSTALLED: 1.2.3 LATEST: 1.2.3c1

and running `pip install --upgrade MySQL-python` fixed my problem.

Cleaning My Data

“So, the basic rule of thumb is, sanitise before use and specifically for that use; not pre-emptively.” - MySQL Documentation

Given the type of data I was working with, I made the elective choice to cleanse my data before trying to insert it into the MYSQL tables rather than once it was stored.

This was done by a combination of research(designing my own advert to see the input limits/types), accounting for white-spaces, unwanted symbols, newlines etc and finally editing strings to suit my needs.

Pythons .strip(), str.replace() and the use of arrays to edit data length worked very well for this and was much faster than mass editing already indexed data.

DataBase Schema

For my overall database design I was left with two main options, a relational DB or a non-relational DB.

After some trial and error, I settled on the non-relational DB for the reasons I will outline below;

Given the structure of my data, and how susceptible it was to dirty reads, I valued DB table integrity far more than a minor speed boost. This meant instead of a multi layer schema i.e one table of models that match’s to tables for mileage, Transmission etc all matched via unique ID. It was safer and more practical to keep a particular car model to a single table and keep the tables separated.

If I then ran into a table issue(which I did), I would only need to re-scrape the single car model I needed as opposed to needing to edit an entire section of a table.

Had my DB been much much bigger, I would have opted for a relational schema but I believe given the data I was working with, I made the right choice from both a management and data integrity stand point.

DataBase Optimisation

To make my DB as effect as possible, it made sense for me to define the types of items being indexed.

My first table consisted of columns of type TEXT.

After designing my own add, I had an understanding of what could possibly be entered into each field I was scraping, this in turn allowed me to design very efficient tables.

ID - Basic ID within table to index car advert

Title - VarChar(255) - A varying string that could be a maximum of 255 characters(Carzone will only allow titles of 200 characters)

Link - VarChar(255) - A varying string that could be a maximum of 255 characters(Carzone will generate links of 150-170 characters)

Price - VarChar(255) - A varying string that could be a maximum of 255 characters, I needed to use a Varchar and not Int to account for the euro symbol and , i.e. 13,000

carYear - SmallInt(50) - A Int that could be a value up to 39,637 and could be 50 characters long

Location - VarChar(255) - 32 possible counties.

Mileage - VarChar(255) - For same reason as price

EngineSize - Float, always a number in decimal format. I.e. 2.0

EngineType - ENUM('Diesel','Petrol') - Can only be one of these two types which is pre-defined so misspelling is impossible. This was far more efficient than a VarChar and also smaller than a SET.

Transmission - ENUM('Manual','Automatic') - Same as above

Colour - VarChar(255)

Owners - TinyInt(50) - A int that could be between 0-127, while this field is left blank for users to decide, it makes no sense for a car to have anything over 10 owners. Hence I chose this column type and outliers will be spotted rather easily.

NCT - VarChar(255) - Always a date but not specified in a format I could use for the DATE type in MySQL.

BodyType - VarChar(255)

Data Mining

Goal

To predict the asking price for a car based on its attributes and then decide whether the asking price is reasonable or not, also my application became rather useful at spotting human error's such as;

Instead of a engine size of 2.0 Litres, a user entered 20 by mistake.

Regression

"In statistics, **regression analysis** is a statistical process for estimating the relationships among variables" - WikaPedia

I chose to Linear Regression to map my model and predict what would be a reasonable asking price given what we know about the rest of the cars. LR was the algorithm of choice as it is the most efficient for analysing my data. Had I had more time, I would have trained a SVM and compared the results as I feel that an SVM may have been more accurate for my type of prediction.

Data To Use

Price - I would keep this separate to my LR algorithm and compare after. The actual data I would use to train my model was the items that were deemed the most important to a car's value - Mileage, Owners, BodyType, NCT(I should have modified this to 3 categories as some car's don't require a NCT and my current model still negatively marks those cars), Colours(Given more time I would have used regex to contain these to 6 main colour groups), Location(given more time I would have used regex to contain these to dublin vs muster vs lenister vs ulster), EngineType, EngineSize, transmission and the age of the car(current year - the cars year).

Algorithm Used

Linear Regression

- http://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html

Language Used

Python/numpy/Pandas/Sklearn/Yhat

Final Implementation

My data mining algorithm works as thus;

Open connection my MQSQL database
Read table into pandas array
Do some string formatting to ensure all elements are useable and make sense
Do the same for my testing file(A file I will feed to the algorithm to test how accurate my model is)
Use fillna to fill in any empty nan, Nan etc
Edit dataframes to drop colums I dont need
Use DictVectoizer to map nw dataframe to bumpy array
Train LinearRegression model without price variable
Use Yhat's basemodel implementation to evaluate LR function
Return predicted prices for each car and finally dump each line into a json file for use later

For the sake of this assignment, I have only provided the result set of one table evaluation(vw_golf scraped data vs predicted price's) but the algorithm can easily be applied to any table within the DB.

Missing Pieces/Misc:

Compare asking price to predicted price in output file(passing that particular parameter was more difficult than expected). Given more time I may have discovered a work around for inserting it into my array but the issue seems to be very common in python.

My prediction accuracy was .70 on average which is good but far from good enough. To increase my accuracy I would need more data and a better model(SVM)

Pieces of my mining algorithm were lacking however and on those 30% of predictions, the price was seen as not accurate at all.

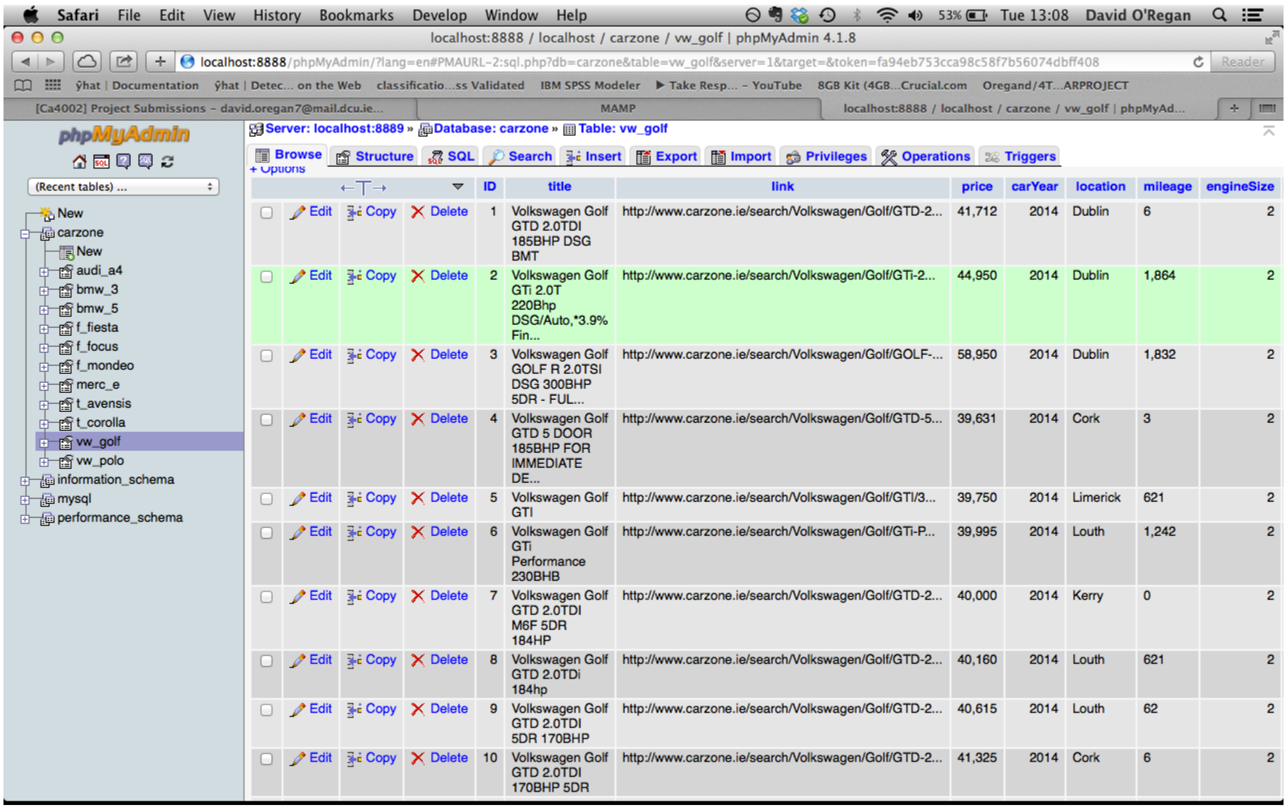
Results?

The results file shows the comparison but on average, most car advertisements overvalued their cars according to my model, which in line with what we know about car sales, is true.

Also, adverts coming directly from dealers had a slightly higher retail price for the same spec car being sold privately, which again makes sense.

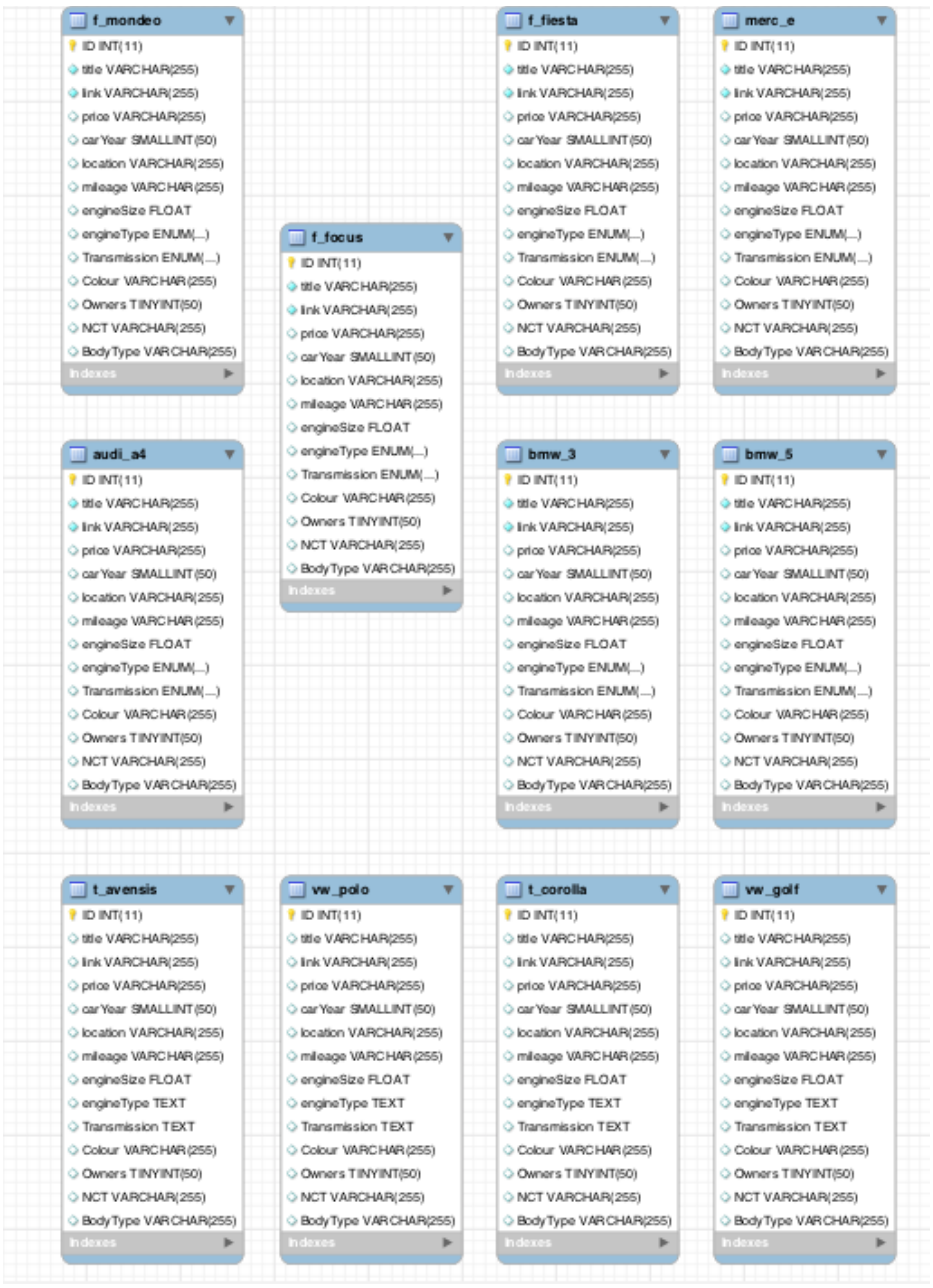
Lastly, when our price prediction is really off(Minus value), this was either due to a user input error OR a issue I didn't have enough time to account for - Car's being negatively marked for empty fields when not needed i.e. A 2014 car with NO mileage/owners/NCT because its brand new, yet my model would mark it negatively even though the information is correct.

DataBase View - phpmyadim


Safari File Edit View History Bookmarks Develop Window Help
Tue 13:08 David O'Regan
localhost:8888 / localhost / carzone / vw_golf | phpMyAdmin 4.1.8
localhost:8888/phpMyAdmin/?lang=en#PMAURL-2:sql.php?db=carzone&table=vw_golf&server=1&target=&token=fa94eb753cca98c58f7b56074dbff408
phpMyAdmin
(Recent tables) ...
New
carzone
New
audi_a4
bmw_3
bmw_5
f_fiesta
f_focus
f_mondeo
merc_e
t_avensis
t_corolla
vw_golf
vw_polo
information_schema
mysql
performance_schema
Server: localhost:8889 » Database: carzone » Table: vw_golf
Browse
Structure
SQL
Search
Insert
Export
Import
Privileges
Operations
Triggers
+ Options

ID	title	link	price	carYear	location	mileage	engineSize
1	Volkswagen Golf GTD 2.0TDI 185BHP DSG BMT	http://www.carzone.ie/search/Volkswagen/Golf/GTD-2...	41,712	2014	Dublin	6	2
2	Volkswagen Golf GTi 2.0T 220Bhp DSG/Auto,*3.9% Fin...	http://www.carzone.ie/search/Volkswagen/Golf/GTi-2...	44,950	2014	Dublin	1,864	2
3	Volkswagen Golf GOLF R 2.0TSI DSG 300BHP 5DR - FUL...	http://www.carzone.ie/search/Volkswagen/Golf/GOLF-...	58,950	2014	Dublin	1,832	2
4	Volkswagen Golf GTD 5 DOOR 185BHP FOR IMMEDIATE DE...	http://www.carzone.ie/search/Volkswagen/Golf/GTD-5...	39,631	2014	Cork	3	2
5	Volkswagen Golf GTI	http://www.carzone.ie/search/Volkswagen/Golf/GTI/3...	39,750	2014	Limerick	621	2
6	Volkswagen Golf GTi Performance 230BHB	http://www.carzone.ie/search/Volkswagen/Golf/GTi-P...	39,995	2014	Louth	1,242	2
7	Volkswagen Golf GTD 2.0TDI M6F 5DR 184HP	http://www.carzone.ie/search/Volkswagen/Golf/GTD-2...	40,000	2014	Kerry	0	2
8	Volkswagen Golf GTD 2.0TDi 184hp	http://www.carzone.ie/search/Volkswagen/Golf/GTD-2...	40,160	2014	Louth	621	2
9	Volkswagen Golf GTD 2.0TDI 5DR 170BHP	http://www.carzone.ie/search/Volkswagen/Golf/GTD-2...	40,615	2014	Louth	62	2
10	Volkswagen Golf GTD 2.0TDI 170BHP 5DR	http://www.carzone.ie/search/Volkswagen/Golf/GTD-2...	41,325	2014	Cork	6	2

Non-Relational DB Schema



f_mondeo
ID INT(11)
title VARCHAR(255)
link VARCHAR(255)
price VARCHAR(255)
carYear SMALLINT(50)
location VARCHAR(255)
mileage VARCHAR(255)
engineSize FLOAT
engineType ENUM(...)
Transmission ENUM(...)
Colour VARCHAR(255)
Owners TINYINT(50)
NCT VARCHAR(255)
BodyType VARCHAR(255)
Indexes
f_focus
ID INT(11)
title VARCHAR(255)
link VARCHAR(255)
price VARCHAR(255)
carYear SMALLINT(50)
location VARCHAR(255)
mileage VARCHAR(255)
engineSize FLOAT
engineType ENUM(...)
Transmission ENUM(...)
Colour VARCHAR(255)
Owners TINYINT(50)
NCT VARCHAR(255)
BodyType VARCHAR(255)
Indexes
f_fiesta
ID INT(11)
title VARCHAR(255)
link VARCHAR(255)
price VARCHAR(255)
carYear SMALLINT(50)
location VARCHAR(255)
mileage VARCHAR(255)
engineSize FLOAT
engineType ENUM(...)
Transmission ENUM(...)
Colour VARCHAR(255)
Owners TINYINT(50)
NCT VARCHAR(255)
BodyType VARCHAR(255)
Indexes
merc_e
ID INT(11)
title VARCHAR(255)
link VARCHAR(255)
price VARCHAR(255)
carYear SMALLINT(50)
location VARCHAR(255)
mileage VARCHAR(255)
engineSize FLOAT
engineType ENUM(...)
Transmission ENUM(...)
Colour VARCHAR(255)
Owners TINYINT(50)
NCT VARCHAR(255)
BodyType VARCHAR(255)
Indexes
audi_a4
ID INT(11)
title VARCHAR(255)
link VARCHAR(255)
price VARCHAR(255)
carYear SMALLINT(50)
location VARCHAR(255)
mileage VARCHAR(255)
engineSize FLOAT
engineType ENUM(...)
Transmission ENUM(...)
Colour VARCHAR(255)
Owners TINYINT(50)
NCT VARCHAR(255)
BodyType VARCHAR(255)
Indexes
bmw_3
ID INT(11)
title VARCHAR(255)
link VARCHAR(255)
price VARCHAR(255)
carYear SMALLINT(50)
location VARCHAR(255)
mileage VARCHAR(255)
engineSize FLOAT
engineType ENUM(...)
Transmission ENUM(...)
Colour VARCHAR(255)
Owners TINYINT(50)
NCT VARCHAR(255)
BodyType VARCHAR(255)
Indexes
bmw_5
ID INT(11)
title VARCHAR(255)
link VARCHAR(255)
price VARCHAR(255)
carYear SMALLINT(50)
location VARCHAR(255)
mileage VARCHAR(255)
engineSize FLOAT
engineType ENUM(...)
Transmission ENUM(...)
Colour VARCHAR(255)
Owners TINYINT(50)
NCT VARCHAR(255)
BodyType VARCHAR(255)
Indexes
t_avensis
ID INT(11)
title VARCHAR(255)
link VARCHAR(255)
price VARCHAR(255)
carYear SMALLINT(50)
location VARCHAR(255)
mileage VARCHAR(255)
engineSize FLOAT
engineType TEXT
Transmission TEXT
Colour VARCHAR(255)
Owners TINYINT(50)
NCT VARCHAR(255)
BodyType VARCHAR(255)
Indexes
vw_polo
ID INT(11)
title VARCHAR(255)
link VARCHAR(255)
price VARCHAR(255)
carYear SMALLINT(50)
location VARCHAR(255)
mileage VARCHAR(255)
engineSize FLOAT
engineType TEXT
Transmission TEXT
Colour VARCHAR(255)
Owners TINYINT(50)
NCT VARCHAR(255)
BodyType VARCHAR(255)
Indexes
t_corolla
ID INT(11)
title VARCHAR(255)
link VARCHAR(255)
price VARCHAR(255)
carYear SMALLINT(50)
location VARCHAR(255)
mileage VARCHAR(255)
engineSize FLOAT
engineType TEXT
Transmission TEXT
Colour VARCHAR(255)
Owners TINYINT(50)
NCT VARCHAR(255)
BodyType VARCHAR(255)
Indexes
vw_golf
ID INT(11)
title VARCHAR(255)
link VARCHAR(255)
price VARCHAR(255)
carYear SMALLINT(50)
location VARCHAR(255)
mileage VARCHAR(255)
engineSize FLOAT
engineType TEXT
Transmission TEXT
Colour VARCHAR(255)
Owners TINYINT(50)
NCT VARCHAR(255)
BodyType VARCHAR(255)
Indexes