



# JAVA 1 팀 중고차검색

전규환, 손병규, 신봉규, 채예진, 박재형

The background of the slide is a dark, textured surface covered with numerous small, grey toy cars, likely from the Hot Wheels brand, arranged in a somewhat chaotic pattern. The cars are viewed from a high angle, and their shadows are cast onto the surface.

# Buy your CAR

1. JAVA 소스코드

2. DB 테이블코드

3. 실행결과

# 중고차를 합리적인 가격으로



중고차 구매 시 꼭 필요한 정보

년식, 주행거리, 색상, 사고유무, 가격, ...

1.  
JAVA 소스코드

1-1.

## MemberDAO

```
package dao;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.ArrayList;

import dto.MemberVO;

public class MemberDAO {
    private ArrayList<MemberVO> dtos;
    private Connection con;
    private Statement st;
    private ResultSet rs;

    private PreparedStatement pstmt;

    public MemberDAO() {
        dtos = new ArrayList<MemberVO>();
        try {
            String user = "system";
```

```
//모두조회
public ArrayList<MemberVO> getAllMembers() {
    String SQL="SELECT * FROM USED CAR";
    try {
        rs=st.executeQuery(SQL);
        while(rs.next()) {
            String company = rs.getString("COMPANY");
            String name = rs.getString("NAME");
            int price = rs.getInt("PRICE");
            int km = rs.getInt("KM");
            String color = rs.getString("COLOR");
            String engine =rs.getString("ENGINE");
            String accident =rs.getString("ACCIDENT");
            String buy_date =rs.getString("BUY_DATE");
            MemberVO VO=new MemberVO(company,name,price,km,color,engine,accident,buy_date);
            dtos.add(VO);
            System.out.println("모두조회 완료");
        }
    } catch (SQLException e) {
        e.printStackTrace();
        System.out.println("데이터베이스 연결 오류:"+e.getMessage());
    }
    return dtos;
}
```

모두조회

### 데이터 추가

```
//데이터 추가
public ArrayList<MemberVO> setMembers(String COMPANY,String NAME,int PRICE,
    int KM,String COLOR,String ENGINE,String ACCIDENT,String DATE) {
    String SQL="INSERT INTO USED CAR VALUES(?, ?, ?, ?, ?, ?, ?, ?, SYSDATE)";
    try {
        pstmt = con.prepareStatement(SQL);
        pstmt.setString(1, COMPANY);
        pstmt.setString(2, NAME);
        pstmt.setInt(3, PRICE);
        pstmt.setInt(4, KM);
        pstmt.setString(5, COLOR);
        pstmt.setString(6, ENGINE);
        pstmt.setString(7, ACCIDENT);
        pstmt.setString(8, DATE);
        pstmt.executeUpdate();
        System.out.println("데이터 추가 완료");
    } catch (SQLException e) {
        e.printStackTrace();
        System.out.println("데이터베이스 연결 오류: "+e.getMessage());
    }
    return dtos;
}
```

### 차량 가격 수정, 차량 삭제

```
//차량 가격 수정
public ArrayList<MemberVO> UpdateCarPrice(String car,int price) {
    String SQL = "UPDATE USED CAR SET price=? where name=?";
    try {
        pstmt = con.prepareStatement(SQL);
        pstmt.setInt(1, price);
        pstmt.setString(2, car);
        pstmt.executeUpdate();
        System.out.println("가격수정완료");
    } catch (SQLException e) {
        e.printStackTrace();
    }
    return dtos;
}

//차량 삭제
public ArrayList<MemberVO> DeleteCar(String car) {
    String SQL = "DELETE FROM USED CAR where name =?";
    try {
        pstmt = con.prepareStatement(SQL);
        pstmt.setString(1, car);
        pstmt.executeUpdate();
        System.out.println("삭제완료");
    } catch (SQLException e) {
        e.printStackTrace();
    }
    return dtos;
}
```



## 1-3. MemberDAO

### 가격 조건검색

```
//가격사이의 가격 검색
public ArrayList<MemberVO> BetweenSearchCar(int min,int max) {
    String SQL = "SELECT * FROM USED CAR WHERE price>="+min+" and price<="+max;
    try {
        pstmt = con.prepareStatement(SQL);
        rs=st.executeQuery(SQL);
        while(rs.next()) {
            String company = rs.getString("COMPANY");
            String name = rs.getString("NAME");
            int price = rs.getInt("PRICE");
            int km = rs.getInt("KM");
            String color = rs.getString("COLOR");
            String engine =rs.getString("ENGINE");
            String accident =rs.getString("ACCIDENT");
            String buy_date =rs.getString("BUY_DATE");
            MemberVO VO=new MemberVO(company,name,price,
            | | | km,color,engine,accident,buy_date);
            dtos.add(VO);
            pstmt.executeUpdate();
        }
        System.out.println("프루트링 완료");
    } catch (SQLException e) {
        e.printStackTrace();
    };
    return dtos;
}
```

단어가 포함된 데이터 검색

```
//LIKE
public ArrayList<MemberVO> SearchCar(String CarName) {
    String SQL = "SELECT * FROM USED CAR WHERE NAME LIKE '%" +CarName+"%'";
    try {
        pstmt = con.prepareStatement(SQL);
        rs=st.executeQuery(SQL);
        while(rs.next()) {
            String company = rs.getString("COMPANY");
            String name = rs.getString("NAME");
            int price = rs.getInt("PRICE");
            int km = rs.getInt("KM");
            String color = rs.getString("COLOR");
            String engine =rs.getString("ENGINE");
            String accident =rs.getString("ACCIDENT");
            String buy_date =rs.getString("BUY_DATE");
            MemberVO VO=new MemberVO(company,name,price,km,color,engine,accident,buy_date);
            dtos.add(VO);
            pstmt.executeUpdate();
        }
    } catch (SQLException e) {
        e.printStackTrace();
    };
    return dtos;
}
```

## 2-1. MemberVO

```
package dto;

public class MemberVO {

    private String company;
    private String name;
    private int price;
    private int km;
    private String color;
    private String engine;
    private String accident;
    private String date;

    public MemberVO(String company,String name,int price,int km,
        String color,String engine,String accident,String date) {
        this.company=company;
        this.name=name;
        this.price=price;
        this.km=km;
        this.color=color;
        this.engine=engine;
        this.accident=accident;
        this.date=date;
    }

    public void setCOMPANY(String company) {
        this.company=company;
    }

    public String getCOMPANY() {
        return company;
    }

    public void setName(String name) {
        this.name=name;
    }

    public String getName() {
        return name;
    }
}
```

```
    public void setPRICE(int price) {
        this.price=price;
    }

    public int getPRICE() {
        return price;
    }

    public void setKM(int km) {
        this.km=km;
    }

    public int getKM() {
        return km;
    }

    public void setColor(String color) {
        this.color=color;
    }

    public String getColor() {
        return color;
    }
}
```

```
    public void setENGINE(String engine) {
        this.engine=engine;
    }

    public String getENGINE() {
        return engine;
    }

    public void setACCIDENT(String accident) {
        this.accident=accident;
    }

    public String getACCIDENT() {
        return accident;
    }

    public void setDate(String date) {
        this.date=date;
    }

    public String getDate() {
        return date;
    }
}
```



### 3-1. MemberService

```
package service;

import java.util.ArrayList;
import dao.MemberDAO;
import dto.MemberVO;

public class MemberService {

    //dao를 포함
    private MemberDAO dao;
    public MemberService() {
        dao=new MemberDAO();
    }

    //모든 조회
    public ArrayList<MemberVO> getAllMembers(){
        return dao.getAllMembers();
    }

    //데이터 추가 "INSERT INTO USED CAR VALUES(?, ?, ?, ?, ?, ?, ?, ?, SYSDATE)"
    public ArrayList<MemberVO> setMembers(String COMPANY,String NAME,int
        PRICE,int KM,String COLOR,String ENGINE,String ACCIDENT,String DATE){
        return dao.setMembers(COMPANY, NAME, PRICE, KM, COLOR, ENGINE, ACCIDENT, DATE);
    }
}
```

```
//차량 가격 변경 "UPDATE USED CAR SET "+car+"="+price+" where NAME = "+car
public ArrayList<MemberVO> UpdateCarPrice(String car,int price){
    return dao.UpdateCarPrice(car,price);
}
```

```
//차량 삭제 "DELETE FROM USED CAR where name = "+car
public ArrayList<MemberVO> DeleteCar(String car) {
    return dao.DeleteCar(car);
}
```

```
//차량가격 조건 검색 "SELECT * FROM USED CAR WHERE price="+min+" and price="+max;
public ArrayList<MemberVO> BetweenSearchCar(int min,int max){
    return dao.BetweenSearchCar(min, max);
}
```

```
//자유조건 검색 "SELECT * FROM USED CAR WHERE NAME LIKE '"+CarName+"%";
public ArrayList<MemberVO> SearchCar(String CarName){
    return dao.SearchCar(CarName);
}
```

```
}
```

## 4-1. HomeController

```
package Controller;

import java.util.ArrayList;
import java.util.Scanner;

import dto.MemberVO;
import service.MemberService;

public class HomeController {

    public static void main(String[] args)
    {
        Scanner sc = new Scanner(System.in);
        ArrayList<MemberVO> dtos;
        MemberService service = new MemberService();
        int choice = 0;
        String company, name, color, engine, accident, buy_date;
        int price, km;
        while(true)
        {
            menu();
            choice = sc.nextInt();
            if(choice==1) { // 전체 검색
                dtos=service.getAllMembers();
                for(int i=0; i<dtos.size(); i++) {
                    System.out.printf("제조사명:%s ", dtos.get(i).getCOMPANY());
                    System.out.printf("차량 모델명:%s ", dtos.get(i).getNAME());
                    System.out.printf("가격:%d원 ", dtos.get(i).getPRICE());
                    System.out.printf("주행거리:%dkm ", dtos.get(i).getKM());
                    System.out.printf("색깔:%s ", dtos.get(i).getColor());
                    System.out.printf("엔진 종류:%s ", dtos.get(i).getENGINE());
                    System.out.printf("사고유무:%s ", dtos.get(i).getACCIDENT());
                    System.out.printf("구매일:%s \n", dtos.get(i).getDate());
                }
                dtos.clear();
                System.out.println("");
            }
        }
    }
}
```

전체검색

select\*from UsedCar;

## 4-2. HomeController

```
else if(choice==2) {
    int min, max;
    System.out.println("최소가격:");
    min=sc.nextInt();
    System.out.print("최대가격:");
    max=sc.nextInt();
    dtos=service.BetweenSearchCar(min, max);
    for(int i=0; i<dtos.size(); i++) {
        System.out.printf("제조사명:%s ", dtos.get(i).getCOMPANY());
        System.out.printf("차량 모델명:%s ", dtos.get(i).getNAME());
        System.out.printf("가격:%d원 ", dtos.get(i).getPRICE());
        System.out.printf("주행거리:%dkm ", dtos.get(i).getKM());
        System.out.printf("색깔:%s ", dtos.get(i).getColor());
        System.out.printf("엔진 종류:%s ", dtos.get(i).getENGINE());
        System.out.printf("사고유무:%s", dtos.get(i).getACCIDENT());
        System.out.printf("구매일:%s\n", dtos.get(i).getDate());
    }
    dtos.clear();
}
else if(choice==3) { //차량 추가
    System.out.print("제조사명 입력:");
    sc.nextLine();
    company=sc.nextLine();
    System.out.print("차량 모델명 입력:");
    name=sc.next();
    System.out.print("가격 입력:");
    price=sc.nextInt();
    System.out.print("주행거리 입력:");
    km=sc.nextInt();
    System.out.print("색깔 입력:");
    color=sc.next();
    System.out.print("엔진종류 입력:");
    engine=sc.next();
    System.out.print("사고유무 입력:");
    accident=sc.next();
    System.out.print("구매일 입력:");
    buy_date=sc.next();
    service.setMembers(company, name, price, km,
        color, engine, accident, buy_date);
}
```

조건검색(between A and B)

select\*from UsedCar where price between 1000 and 2000;

데이터 추가

insert into UsedCar values ('현대','악센트',930,28089,'주색',  
'가솔린','N','2017-07-13');

## 4-2. HomeController

```
else if(choice==4) {
    System.out.print("차량 모델명 입력:");
    sc.nextLine();
    name=sc.nextLine();
    System.out.print("수정할 가격 입력:");
    price=sc.nextInt();
    service.UpdateCarPrice(name, price);
}
else if(choice==5) {
    System.out.print("차량 모델명 입력:");
    name=sc.next();
    service.DeleteCar(name);
}
else if(choice==6) {
    System.out.println("차량 모델명 입력:");
    name=sc.next();
    dtos=service.SearchCar(name);
    for(int i=0; i<dtos.size(); i++) {
        System.out.printf("제조사명:%s ", dtos.get(i).getCOMPANY());
        System.out.printf("차량 모델명:%s ", dtos.get(i).getName());
        System.out.printf("가격:%d원 ", dtos.get(i).getPRICE());
        System.out.printf("주행거리:%dkm ", dtos.get(i).getKM());
        System.out.printf("색깔:%s ", dtos.get(i).getColor());
        System.out.printf("엔진 종류:%s ", dtos.get(i).getENGINE());
        System.out.printf("사고유무:%s", dtos.get(i).getACCIDENT());
        System.out.printf("구매일:%s\n", dtos.get(i).getDate());
    }
    dtos.clear();
}
else if(choice==7) {
    break;
}
}
```

### 데이터 갱신

Update UsedCar set price=1500 where name='아반떼';

### 데이터 삭제

Delete from UsedCar where name='K5';

### 단어가 포함된 데이터 검색

Select\*from UsedCar where name like "%"+CarName+"%";

```
public static void menu() {
    System.out.println("1. 모든 중고차 조회");
    System.out.println("2. 조건 검색");
    System.out.println("3. 중고차 추가");
    System.out.println("4. 중고차 수정");
    System.out.println("5. 중고차 삭제");
    System.out.println("6. 차량 모델명 검색");
    System.out.println("7. 종료");
}
```

## 2. DB 테이블 코드



## 1-1. DB table

```
SQL> create table UsedCar(  
2  company varchar (20) NOT NULL,  
3  name varchar2(20) NOT NULL,  
4  price int NOT NULL,  
5  km int NOT NULL,  
6  color varchar2(20) NOT NULL,  
7  engine varchar2(20) NOT NULL,  
8  accident varchar2(20) NOT NULL,  
9  constraint accidentr_ck check (accident in('y','n')),  
10 buy_date date NOT NULL  
11 );
```

테이블이 생성되었습니다.

```
SQL> _
```

## 테이블 생성

## 데이터 삽입

```
insert into UsedCar values ('르노삼성', 'SM3', 1250, 25938, '은색', '가솔린', 'n', '2019-05-23');  
insert into UsedCar values ('현대', '악센트', 930, 28089, '파랑', '가솔린', 'n', '2017-07-13');  
insert into UsedCar values ('기아', '레뉴모닝', 750, 32689, '베이지', '가솔린', 'n', '2017-01-05');  
insert into UsedCar values ('현대', '코나', 2020, 11755, '파랑', '디젤', 'n', '2018-01-02');  
insert into UsedCar values ('기아', '스포티지', 1570, 47260, '은색', '디젤', 'n', '2016-08-05');  
insert into UsedCar values ('기아', 'K7', 2690, 36741, '검정색', 'LPG', 'n', '2018-02-23');  
insert into UsedCar values ('GM대우', '다마스', 450, 39809, '파랑', 'LPG', 'n', '2015-11-16');  
insert into UsedCar values ('현대', '그랜저', 1590, 84167, '검정색', 'LPG', 'n', '2014-07-08');  
insert into UsedCar values ('르노삼성', 'QMG', 2900, 11, '파랑', 'LPG', 'n', '2021-05-14');  
insert into UsedCar values ('현대', '아반떼', 2560, 8604, '파랑', '하이브리드', 'y', '2021-04-02');  
insert into UsedCar values ('기아', 'K5', 920, 112281, '파랑', '하이브리드', 'n', '2012-11-28');  
insert into UsedCar values ('기아', '쏘렌토', 4600, 19, '검정색', '하이브리드', 'n', '2020-08-10');  
insert into UsedCar values ('쌍용', '코란도', 2140, 41945, '파랑', '디젤', 'n', '2019-08-30');  
insert into UsedCar values ('현대', '포터', 1190, 124316, '파랑', '디젤', 'n', '2015-09-10');  
insert into UsedCar values ('벤츠', 'GLA클래스', 5200, 9664, '파랑', '가솔린', 'n', '2020-11-24');  
insert into UsedCar values ('쌍용', '베리뉴티볼러', 2280, 2124, '파랑', '가솔린', 'n', '2020-03-04');  
insert into UsedCar values ('제네시스', 'G70', 3650, 25237, '회색', '가솔린', 'n', '2019-02-19');  
insert into UsedCar values ('BMW', 'F30', 1620, 107650, '파랑', '디젤', 'n', '2014-08-19');  
insert into UsedCar values ('현대', 'YF쏘나타', 380, 199724, '검정', '가솔린', 'n', '2009-12-23');  
insert into UsedCar values ('제네시스', 'G70', 3520, 66068, '검정', '가솔린', 'n', '2018-05-08');  
insert into UsedCar values ('쉐보레', '대우 로스카', 109, 166642, '검정', '가솔린', 'y', '2006-05-04');  
insert into UsedCar values ('르노삼성', 'SM5뉴임프레션', 150, 250735, '은색', '가솔린', 'n', '2007-07-23');  
insert into UsedCar values ('포르쉐', '카이엔 (2세대)', 7630, 93247, '파랑', '가솔린', 'n', '2021-03-26');  
insert into UsedCar values ('BMW', 'X4M', 8200, 19100, '파랑', '가솔린', 'n', '2020-05-29');  
insert into UsedCar values ('랜드로버', '레인지로버', 17950, 23000, '파랑', '가솔린', 'n', '2018-11-21');
```

### 3. 실행결과

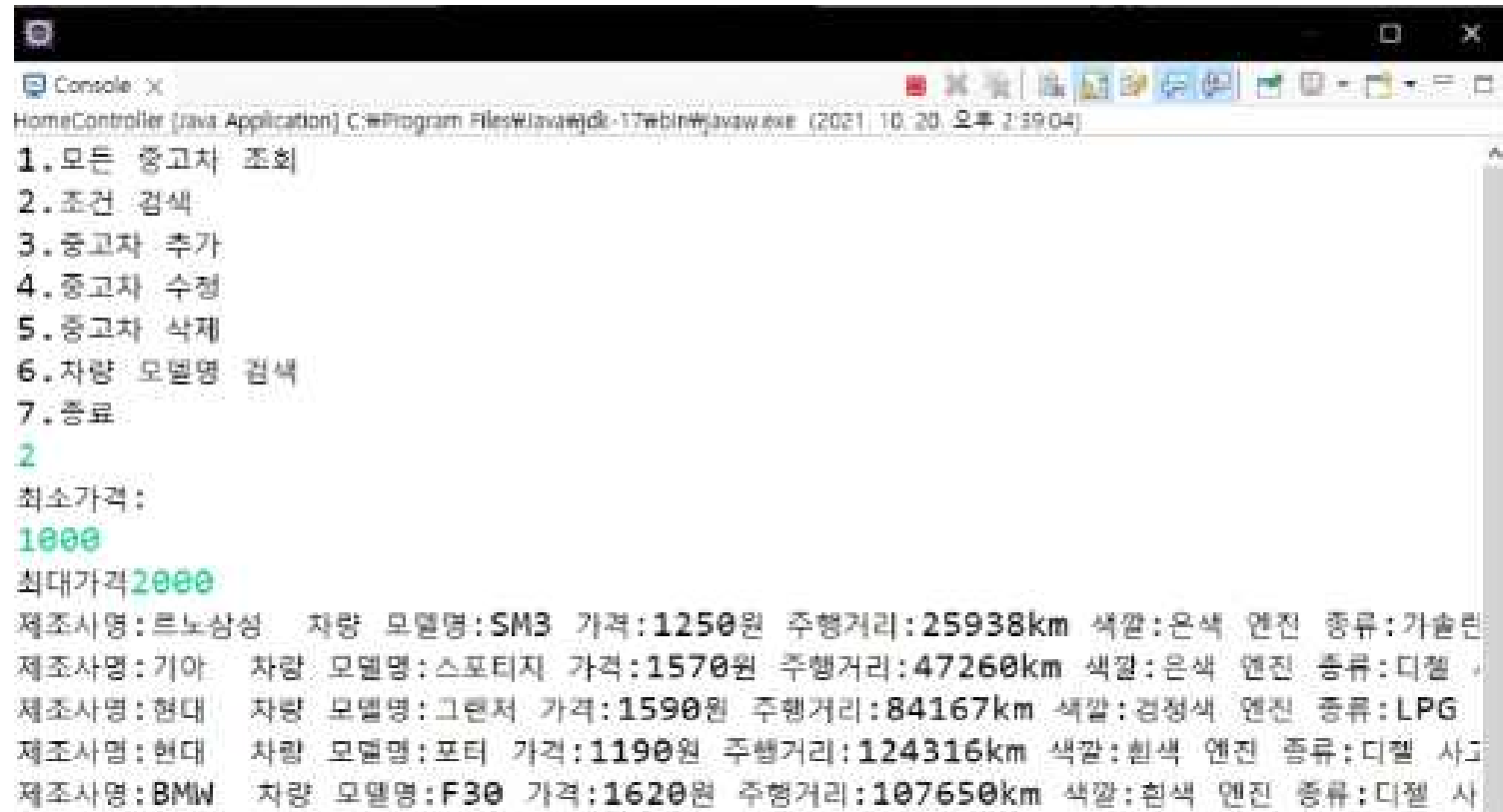
## 1-1. JAVA 실행화면



```
Console x
HomeController (Java Application)
1. 모든 중고차 조회
2. 조건 검색
3. 중고차 추가
4. 중고차 수정
5. 중고차 삭제
6. 차량 모델명 검색
7. 종료
1
제조사명:르노삼성 차량 모델명:SM3 가격:1250원 주행거리:25938km 색깔:은색 엔진 종류:가솔린
제조사명:현대 차량 모델명:악센트 가격:930원 주행거리:28089km 색깔:좌색 엔진 종류:가솔린 사
제조사명:기아 차량 모델명:더뉴모닝 가격:750원 주행거리:32689km 색깔:베이지 엔진 종류:가솔린
제조사명:현대 차량 모델명:코나 가격:2020원 주행거리:11755km 색깔:흰색 엔진 종류:디젤 사고
제조사명:기아 차량 모델명:스포티지 가격:1570원 주행거리:47260km 색깔:은색 엔진 종류:디젤
제조사명:기아 차량 모델명:K7 가격:2690원 주행거리:36741km 색깔:검정색 엔진 종류:LPG 사고
제조사명:GM대우 차량 모델명:다마스 가격:450원 주행거리:39809km 색깔:청색 엔진 종류:LPG
제조사명:현대 차량 모델명:그랜저 가격:1590원 주행거리:84167km 색깔:거치색 엔진 종류:LPG
```

모두 조회 : `Select*from UsedCar;`

## 1-2. JAVA 실행화면



```
Console
HomeController [Java Application] C:\Program Files\Java\jdk-17\bin\javaw.exe (2021. 10. 20. 오후 2:39:04)

1. 모든 중고차 조회
2. 조건 검색
3. 중고차 추가
4. 중고차 수정
5. 중고차 삭제
6. 차량 모델명 검색
7. 종료

2
최소가격:
1000
최대가격2000
제조사명:르노삼성 차량 모델명:SM3 가격:1250원 주행거리:25938km 색깔:은색 엔진 종류:가솔린
제조사명:기아 차량 모델명:스포티지 가격:1570원 주행거리:47260km 색깔:은색 엔진 종류:디젤
제조사명:현대 차량 모델명:그랜저 가격:1590원 주행거리:84167km 색깔:검정색 엔진 종류:LPG
제조사명:현대 차량 모델명:포터 가격:1190원 주행거리:124316km 색깔:흰색 엔진 종류:디젤 사
제조사명:BMW 차량 모델명:F30 가격:1620원 주행거리:107650km 색깔:흰색 엔진 종류:디젤 사
```

### 조건 검색

Select\*from UsedCar where price between 1000 and 2000;

Select\*from UsedCar where price>=1000 and price<=2000;

## 1-3. JAVA 실행화면

1. 모든 중고차 조회
2. 조건 검색
3. 중고차 추가
4. 중고차 수정
5. 중고차 삭제
6. 차량 모델명 검색
7. 종료

6

차량 모델명 입력:

그랜저

제조사명:현대 차량 모델명:그랜저 가격:1590원 주행거리:84167km 색깔:검정색 엔진 종류:LPG

1. 모든 중고차 조회
2. 조건 검색
3. 중고차 추가
4. 중고차 수정
5. 중고차 삭제
6. 차량 모델명 검색
7. 종료

조건검색

Select\*from UsedCar where name like '그랜저';





감사합니다