오라클 한글 초성 검색하기

SELECT \* FROM ajax\_book

WHERE b\_title BETWEEN NCHR(51088) AND NCHR(52263)

1. 초성은 19자로

ㄱ, ㄲ, ㄴ, ㄷ, ㄸ, ㄹ, ㅁ, ㅂ, ㅃ, ㅅ,

ㅆ, ㅇ, ㅈ, ㅉ, ㅊ, ㅋ, ㅌ, ㅍ, ㅎ

2. 중성은 21자로

ㅏ, ㅐ, ㅑ, ㅒ, ㅓ, ㅔ, ㅕ, ㅖ, ㅗ, ㅘ,

ㅙ, ㅚ, ㅛ, ㅜ, ㅝ, ㅞ, ㅟ, ㅠ, ㅡ, ㅢ,

ㅣ

3. 종성은 27자로

ㄱ, ㄲ, ㄳ, ㄴ, ㄵ, ㄶ, ㄷ, ㄹ, ㄺ,

ㄻ, ㄼ, ㄽ, ㄾ, ㄿ, ㅀ, ㅁ, ㅂ, ㅄ, ㅅ,

ㅆ, ㅇ, ㅈ, ㅊ, ㅋ, ㅌ, ㅍ, ㅎ

< 규칙1>

1.중성이 바뀌는 경우 코드값 차이 : 28

2.초성이 바뀌는 경우 코드값 차이 : 588(21\*28)

3. 한글에 대한 첫 코드값('가'의 코드값): 44032

<응용>

가 - 44032

까 - 44620

나 - 45208

다 - 45796

따 - 46384

라 - 46972

마 - 47560

바 - 48148

빠 - 48736

사 - 49324

싸 - 49912

아 - 50500

자 - 51088

짜 - 51676

차 - 52264

카 - 52852

타 - 53440

파 - 54028

하 - 54616

초성: 글자의 코드에서 44032를 빼고, 21\*28로 나눈 몫 0 ~ 18

중성: 글자의 코드에서 44032를 빼고, 21\*28로 나눈 나머지를 다시 28로 나눈 몫 0 ~ 20

종성: 글자의 코드에서 44032를 빼고, 21\*28로 나눈 나머지를 다시 28로 나눈 나머지 0 ~ 27

function choCheck(str)

{

var cho = ["ㄱ","ㄲ","ㄴ","ㄷ","ㄸ","ㄹ","ㅁ","ㅂ","ㅃ","ㅅ","ㅆ","ㅇ"

,"ㅈ","ㅉ","ㅊ","ㅋ","ㅌ","ㅍ","ㅎ"];

var code;

//alert("가".charCodeAt(0));//?

code = "정".charCodeAt(0)-44032;//7189

alert(Math.floor(code/(21\*28)));//12

alert("code:"+code);

result = cho[12];

alert("result"+cho[12]);//ㅈ

code = str.charCodeAt(0);

}

  function choHangul(str){

  //alert("str:"+str);

  cho = ["ㄱ","ㄲ","ㄴ","ㄷ","ㄸ","ㄹ","ㅁ","ㅂ","ㅃ","ㅅ","ㅆ","ㅇ","ㅈ","ㅉ","ㅊ","ㅋ","ㅌ","ㅍ","ㅎ"];

  result ="";

  for(i=0;i<str.length;i++){

  //alert("길이:"+str.length);

  code = str.charCodeAt(i)-44032;

  //alert("code:"+code);

  if(code>-1 && code<11172) result += cho[Math.floor(code/588)];

  }

  return result;

  }

  function b\_search(){

  //alert("vb\_title : "+$("#sb\_title").val());

  var vb\_title = $("#sb\_title").val().toUpperCase();

  var choKeyword = choHangul($("#sb\_title").val());

  //초성 검색 구분

  if(vb\_title!="" && choKeyword==""){

  choKeyword = vb\_title;

  choMode ="Y";

  }

  else{

  choKeyword = vb\_title;

  choMode="N";

  }

  //alert("제목:"+$("#sb\_title").val());

  //var vb\_title = $("#sb\_title").val();

  var vb\_author = $("#sb\_author").val();

  var vb\_date = $("#sb\_date").val();

  alert("choMode: "+choMode+"choKeyword: "+choKeyword);

$("#tb\_bookList").datagrid({

url:"./getBookList.bk?b\_title="+choKeyword

  +"&b\_author="+vb\_author

  +"&b\_date="+vb\_date

  +"&choMode="+choMode

});

  }

<select id="getBookList" parameterType="java.util.HashMap" resultType="com.vo.BookVO">

SELECT b\_no, b\_title, b\_img

,b\_date, b\_author, b\_company

FROM ajax\_book

  <where>

<!-- <if test="word!=null">

AND b\_title LIKE '%'||#{word}||'%'

</if> -->

<if test="b\_no!=null">

AND b\_no =#{b\_no}

</if>

<!-- <if test="b\_title!=null">

AND b\_title LIKE '%'||#{b\_title}||'%'

</if> -->

<if test='b\_title!=null and choMode.equals("N")'>

AND b\_title LIKE '%'||#{b\_title}||'%'

</if>

<if test='b\_title!=null and choMode.equals("Y")'>

AND fn\_korinitialkeyword(b\_title) LIKE '%'||#{b\_title}||'%'

</if>

<!-- <if test="b\_author!=null">

AND b\_author LIKE '%'||#{b\_author}||'%'

</if>

<if test="b\_date!=null">

AND b\_date LIKE '%'||#{b\_date}||'%'

</if> -->

</where>

</select>

create or replace function fn\_korinitialkeyword( str in varchar2) return varchar2

is

  returnStr varchar2(100);

  cnt number := 0;

  tmpStr varchar2(10);

begin

  if str is null then

  return '';

  end if;

  cnt := length(str);

  for i in 1 .. cnt

  loop

  tmpStr := substr(str,i,1);

  returnStr := returnStr ||

  case when tmpStr < 'ㄱ' then substr(tmpStr, 1, 1)

  when ascii('ㄱ') <= ascii(tmpStr) and ascii(tmpStr) <= ascii('ㅎ') then chr(ascii(tmpStr))

  when tmpStr < '나' then 'ㄱ'

  when tmpStr < '다' then 'ㄴ'

  when tmpStr < '라' then 'ㄷ'

  when tmpStr < '마' then 'ㄹ'

  when tmpStr < '바' then 'ㅁ'

  when tmpStr < '사' then 'ㅂ'

  when tmpStr < '아' then 'ㅅ'

  when tmpStr < '자' then 'ㅇ'

  when tmpStr < '차' then 'ㅈ'

  when tmpStr < '카' then 'ㅊ'

  when tmpStr < '타' then 'ㅋ'

  when tmpStr < '파' then 'ㅌ'

  when tmpStr < '하' then 'ㅍ'

  else 'ㅎ'

  end;

  end loop;

  return returnStr;

end;

SELECT fn\_korinitialkeyword('자')  --ㅈ

             ,fn\_korinitialkeyword('자석') -- ㅈㅅ

             ,fn\_korinitialkeyword('자석이불') --ㅈㅅㅇㅂ

  FROM dual

SELECT substr('자석',1,1) --자

  ,substr('자석이불',2,1) --석

  ,substr('자석이불',2,2) --석이

  ,substr('자석이불',2,3) --석이불

  ,substr('자석이불',0,1) --자

  FROM dual

SELECT ascii('ㄱ') --42145

  , chr(ascii('ㄱ')) --ㄱ

  , ascii('ㄲ') --42146

  , chr(ascii('ㄲ')) --ㄲ

  , ascii('ㄳ') --42147

  , ascii('ㄴ') --42148

  , ascii('ㄵ') --42149

  , ascii('ㄶ') --42150

  , ascii('나') --45994

  , ascii('노') --46059

  , ascii('ㄷ') --42151

  , ascii('ㅍ') --42173

  , ascii('ㅎ') --42174

  FROM dual

SELECT '123으로 나눴다', FN\_GET\_DIV\_KO\_CHAR('123으로 나눴다') FROM dual

SELECT '밝은 형광등', FN\_GET\_DIV\_KO\_CHAR('밝은 형광등') FROM dual

SELECT \* FROM nls\_database\_parameters

WHERE parameter = 'NLS\_CHARACTERSET'

CREATE OR REPLACE FUNCTION FN\_GET\_DIV\_KO\_CHAR (

    i\_p1    IN    VARCHAR2

)

    RETURN VARCHAR2

AS

    l\_rt    VARCHAR2 (4000);

    FUNCTION f1 (

        i\_p1    IN    VARCHAR2

    )

        RETURN VARCHAR2

    AS

        l\_p1    PLS\_INTEGER     := ASCII (UNISTR (i\_p1));

        l\_i0    PLS\_INTEGER     := l\_p1 - TO\_NUMBER ('AC00', 'XXXX');

        l\_i1    PLS\_INTEGER     := TO\_NUMBER ('1100', 'XXXX') + FLOOR (l\_i0 / (21 \* 28));

        l\_i2    PLS\_INTEGER     := TO\_NUMBER ('1161', 'XXXX') + FLOOR (MOD (l\_i0, (21 \* 28)) / 28);

        l\_i3    PLS\_INTEGER     := TO\_NUMBER ('11A7', 'XXXX') + FLOOR (MOD (l\_i0, 28));

        l\_rt    VARCHAR2(4000);

    BEGIN

        CASE

            WHEN l\_p1 BETWEEN TO\_NUMBER ('AC00', 'XXXX') AND TO\_NUMBER ('D7A3', 'XXXX')

            THEN

                l\_rt := NCHR (l\_i1)

                     || CASE WHEN l\_i2 <> 4519 THEN NCHR (l\_i2) END

                     || CASE WHEN l\_i3 <> 4519 THEN NCHR (l\_i3) END

                     ;

            ELSE

                l\_rt := i\_p1;

        END CASE;

        RETURN l\_rt;

    END f1;

    FUNCTION f2 (

        i\_p1    IN    VARCHAR2

    )

        RETURN VARCHAR2

    AS

        l\_p1    VARCHAR2(10);

        l\_rt    VARCHAR2(30);

    BEGIN

        FOR i IN 1.. LENGTH( i\_p1 )

        LOOP

            l\_p1 := SUBSTR(i\_p1, i, 1);

            IF l\_p1 = 'ᆨ' OR l\_p1 = 'ᄀ' THEN l\_p1 := 'ㄱ';

            ELSIF l\_p1 = 'ᆫ' OR l\_p1 = 'ᄂ' THEN l\_p1 := 'ㄴ';

            ELSIF l\_p1 = 'ᆮ' OR l\_p1 = 'ᄃ' THEN l\_p1 := 'ㄷ';

            ELSIF l\_p1 = 'ᆯ' OR l\_p1 = 'ᄅ' THEN l\_p1 := 'ㄹ';

            ELSIF l\_p1 = 'ᆷ' OR l\_p1 = 'ᄆ' THEN l\_p1 := 'ㅁ';

            ELSIF l\_p1 = 'ᆸ' OR l\_p1 = 'ᄇ' THEN l\_p1 := 'ㅂ';

            ELSIF l\_p1 = 'ᆺ' OR l\_p1 = 'ᄉ' THEN l\_p1 := 'ㅅ';

            ELSIF l\_p1 = 'ᆼ' OR l\_p1 = 'ᄋ' THEN l\_p1 := 'ㅇ';

            ELSIF l\_p1 = 'ᆽ' OR l\_p1 = 'ᄌ' THEN l\_p1 := 'ㅈ';

            ELSIF l\_p1 = 'ᆾ' OR l\_p1 = 'ᄎ' THEN l\_p1 := 'ㅊ';

            ELSIF l\_p1 = 'ᆿ' OR l\_p1 = 'ᄏ' THEN l\_p1 := 'ㅋ';

            ELSIF l\_p1 = 'ᇀ' OR l\_p1 = 'ᄐ' THEN l\_p1 := 'ㅌ';

            ELSIF l\_p1 = 'ᇁ' OR l\_p1 = 'ᄑ' THEN l\_p1 := 'ㅍ';

            ELSIF l\_p1 = 'ᇂ' OR l\_p1 = 'ᄒ' THEN l\_p1 := 'ㅎ';

            ELSIF l\_p1 = 'ᆩ' OR l\_p1 = 'ᄁ' THEN l\_p1 := 'ㄲ';

            ELSIF l\_p1 = 'ᄄ' THEN l\_p1 := 'ㄸ';

            ELSIF l\_p1 = 'ᄈ' THEN l\_p1 := 'ㅃ';

            ELSIF l\_p1 = 'ᄊ' OR l\_p1 = 'ᆻ' THEN l\_p1 := 'ㅆ';

            ELSIF l\_p1 = 'ᄍ' THEN l\_p1 := 'ㅉ';

            ELSIF l\_p1 = 'ᆪ' THEN l\_p1 := 'ᄀᄉ';

            ELSIF l\_p1 = 'ᆬ' THEN l\_p1 := 'ᄂᄌ';

            ELSIF l\_p1 = 'ᆭ' THEN l\_p1 := 'ᄂᄒ';

            ELSIF l\_p1 = 'ᆰ' THEN l\_p1 := 'ᄅᄀ';

            ELSIF l\_p1 = 'ᆱ' THEN l\_p1 := 'ᄅᄆ';

            ELSIF l\_p1 = 'ᆲ' THEN l\_p1 := 'ᄅᄇ';

            ELSIF l\_p1 = 'ᆳ' THEN l\_p1 := 'ᄅᄉ';

            ELSIF l\_p1 = 'ᆴ' THEN l\_p1 := 'ᄅᄐ';

            ELSIF l\_p1 = 'ᆶ' THEN l\_p1 := 'ᄅᄒ';

            ELSIF l\_p1 = 'ᆹ' THEN l\_p1 := 'ᄇᄉ';

            ELSIF l\_p1 = 'ᅡ' THEN l\_p1 := 'ㅏ';

            ELSIF l\_p1 = 'ᅣ' THEN l\_p1 := 'ㅑ';

            ELSIF l\_p1 = 'ᅥ' THEN l\_p1 := 'ㅓ';

            ELSIF l\_p1 = 'ᅧ' THEN l\_p1 := 'ㅕ';

            ELSIF l\_p1 = 'ᅩ' THEN l\_p1 := 'ㅗ';

            ELSIF l\_p1 = 'ㅛ' THEN l\_p1 := 'ㅛ';

            ELSIF l\_p1 = 'ᅮ' THEN l\_p1 := 'ㅜ';

            ELSIF l\_p1 = 'ᅲ' THEN l\_p1 := 'ㅠ';

            ELSIF l\_p1 = 'ᅳ' THEN l\_p1 := 'ㅡ';

            ELSIF l\_p1 = 'ᅵ' THEN l\_p1 := 'ㅣ';

            ELSIF l\_p1 = 'ᅢ' THEN l\_p1 := 'ㅐ';

            ELSIF l\_p1 = 'ᅤ' THEN l\_p1 := 'ㅒ';

            ELSIF l\_p1 = 'ᅦ' THEN l\_p1 := 'ㅔ';

            ELSIF l\_p1 = 'ᅨ' THEN l\_p1 := 'ㅖ';

            ELSIF l\_p1 = 'ᅬ' THEN l\_p1 := 'ㅚ';

            ELSIF l\_p1 = 'ᅱ' THEN l\_p1 := 'ㅟ';

            ELSIF l\_p1 = 'ᅴ' THEN l\_p1 := 'ㅢ';

            END IF;

            l\_rt := l\_rt || l\_p1;

        END LOOP;

        RETURN l\_rt;

    END f2;

BEGIN

    FOR i IN 1..LENGTH( i\_p1 )

    LOOP

        l\_rt := l\_rt || f2( f1( SUBSTR(i\_p1, i, 1) ) );

    END LOOP;

    RETURN l\_rt;

END FN\_GET\_DIV\_KO\_CHAR;