```
-------------------------------------------
--EXECUTE IMMEDIATE - Sorgu
-------------------------------------------

declare
    v_ad varchar2(50);
    v_maas number;

begin

    execute immediate
      'select ad, maas from personel where personel_id = :b1'
        into v_ad, v_maas using 5020;

    dbms_output.put_line(v_ad || ': '||v_maas);

end;

-------------------------------------------

function personel_getir(p_id number) return personel%rowtype
is
    v_sql varchar2(500) := 'select * from personel where personel_id = :b1';
    v_persrow personel%rowtype;
begin

    execute immediate v_sql into v_persrow using p_id;
    return v_persrow;

end;

declare
    v_persrow personel%rowtype;
begin

    v_persrow := PCK_GENEL.PERSONEL_GETIR(5015);
    dbms_output.put_line(v_persrow.ad || ', ' || v_persrow.unvan);

end;

-------------------------------------------
--EXECUTE IMMEDIATE - DML
-------------------------------------------

begin

    execute immediate
      'update personel set maas = maas * :p1 where unvan = :b2'
        using 1.15, 'UZMAN';

    dbms_output.put_line(sql%rowcount);

end;

-------------------------------------------

begin

    execute immediate 'insert into departman values(:1, :2)'
        using 115, 'Dijital dönüşüm Ofisi';

end;
```

```
-------------------------------------------
--EXECUTE IMMEDIATE - DDL
-------------------------------------------

begin

    execute immediate 'create table is_ilanlari (id number)';

    execute immediate 'alter table is_ilanlari add baslik varchar2(25)';

    execute immediate 'truncate table is_ilanlari';

end;

-------------------------------------------
--Data Dictionary ile Dinamik SQL Yazma
-------------------------------------------

SELECT
    'ALTER TABLE ' || TABLE_NAME ||
    ' RENAME TO '  || 'T_' ||TABLE_NAME||';' AS SCRIPT
FROM USER_TABLES;

-------------------------------------------

SELECT 'ALTER TABLE ' || TABLE_NAME ||
    ' ADD ID NUMBER;' AS SCRIPT FROM
(
    SELECT TABLE_NAME FROM USER_TABLES
    MINUS
    SELECT TABLE_NAME FROM USER_TAB_COLUMNS
    WHERE COLUMN_NAME = 'ID'
)

-------------------------------------------
--EXECUTE IMMEDIATE - Dinamik PL/SQL
-------------------------------------------

function yillik_maas(p_id number) return number
is
    v_plsql varchar2(500) :=
        'declare ' ||
        'v_persrow personel%rowtype; ' ||
        'begin ' ||
        'v_persrow := pck_genel.personel_getir(:persid); ' ||
        ':sonuc := v_persrow.maas * 12; ' ||
        'end;';
    v_sonuc number;
begin

    dbms_output.put_line(v_plsql);
    execute immediate v_plsql using in p_id, out v_sonuc;
    return v_sonuc;

end;

exec dbms_output.put_line(pck_genel.yillik_maas(5015));
```

```
-----------------------------------------
--BULK COLLECT INTO
-----------------------------------------

set serveroutput on;
declare
    type t_ad   is table of varchar2 (20);
    type t_maas is table of number;

    v_ad   t_ad;
    v_maas t_maas;
Begin

    select ad, maas
        bulk collect into v_ad, v_maas
    from personel;

    for idx in 1..v_ad.count
    loop
        dbms_output.put_line (idx||' - '||v_ad (idx) ||': '||v_maas (idx));
    end loop;

end;

-----------------------------------------
--BULK COLLECT INTO - Limit
-----------------------------------------

declare
  cursor cur_personel is
     select * from personel;

    type t_personel is table of cur_personel%rowtype;

    v_per_dizi t_personel;
begin
    open cur_personel;

        fetch cur_personel
            bulk collect into v_per_dizi limit 50;

        for indx in 1 ..v_per_dizi.count
        loop
            dbms_output.put_line('Adı:' || v_per_dizi(indx).ad ||
                                 ' Soyadı:' || v_per_dizi(indx).soyad);
        end loop;

   close cur_personel;
end;
```

```
-----------------------------------------
--BULK COLLECT INTO - FORALL
-----------------------------------------

declare
  cursor cur_personel is
     select * from personel;

    type t_personel is table of cur_personel%rowtype;

    v_per_dizi t_personel;
begin
    open cur_personel;

        fetch cur_personel
            bulk collect into v_per_dizi limit 10;

        forall indx in v_per_dizi.first..v_per_dizi.last
            update personel set prim=111 where personel_id = v_per_dizi(indx).personel_id;

   close cur_personel;
end;

-----------------------------------------
--OPEN FOR İfadesi
-----------------------------------------

set serveroutput on;
declare
    type t_refc is ref cursor;
    type t_ad   is table of varchar2 (20);
    type t_maas is table of number;

    c_pers t_refc;
    v_ad   t_ad;
    v_maas t_maas;

begin

    open c_pers for 'select ad, maas from personel';
    fetch c_pers bulk collect into v_ad, v_maas;
    close c_pers;

    for idx in 1..v_ad.count
    loop
        dbms_output.put_line (idx||' - '||v_ad (idx) ||': '||v_maas (idx));
    end loop;

end;
```

```
-----------------------------------------
--DBMS_SQL Örnek-1
-----------------------------------------

function tum_kayitlari_sil(p_tablo_ismi varchar2) return number
is
    v_cur_id pls_integer;
    v_del_rows number;
    v_sql varchar2(100);

begin

    v_sql := 'delete from ' || p_tablo_ismi;
    v_cur_id := DBMS_SQL.OPEN_CURSOR;
    DBMS_SQL.PARSE(v_cur_id, v_sql, DBMS_SQL.NATIVE);
    v_del_rows := DBMS_SQL.EXECUTE(v_cur_id);
    DBMS_SQL.CLOSE_CURSOR(v_cur_id);
    return v_del_rows;

end;

--create table departman_temp as select * from departman;

declare
    v_tablo varchar2(20) := 'DEPARTMAN_TEMP';
    v_silinen_kayit integer := 0;

begin

    v_silinen_kayit := pck_genel.tum_kayitlari_sil(v_tablo);
    dbms_output.put_line( v_tablo || ' tablosundan ' || v_silinen_kayit ||
    ' adet kayıt silinmiştir');

end;
```

```
-------------------------------------------
--DBMS_SQL Örnek-2
-------------------------------------------

procedure is_ilani_ekle(
    p_ilan_id integer,
    p_baslik varchar2,
    p_tarih date,
    p_platform varchar2 default 'Linkedin')
is
    v_cur_id pls_integer;
    v_sql varchar2(100);
    v_rows integer;
begin

    v_sql := 'insert into is_ilanlari values (:bid, :bbaslik, :btarih, :bplatform)';
    v_cur_id := DBMS_SQL.OPEN_CURSOR;
    DBMS_SQL.PARSE(v_cur_id, v_sql, DBMS_SQL.NATIVE);
    DBMS_SQL.BIND_VARIABLE(v_cur_id, ':bid', p_ilan_id);
    DBMS_SQL.BIND_VARIABLE(v_cur_id, ':bbaslik', p_baslik);
    DBMS_SQL.BIND_VARIABLE(v_cur_id, ':btarih', p_tarih);
    DBMS_SQL.BIND_VARIABLE(v_cur_id, ':bplatform', p_platform);
    v_rows := DBMS_SQL.EXECUTE(v_cur_id);
    DBMS_SQL.CLOSE_CURSOR(v_cur_id);

end;

--create sequence sq_is_ilani start with 1 increment by 1;

begin
    PCK_GENEL.IS_ILANI_EKLE(1, 'PL/SQL Developer', sysdate+2, 'Kariyer');
    PCK_GENEL.IS_ILANI_EKLE(2, 'Oracle DBA', sysdate+1);
    PCK_GENEL.IS_ILANI_EKLE(sq_is_ilani.nextval, 'Senior Java Developer', sysdate+5);
end;

-------------------------------------------
--DBMS_SQL Örnek-3
-------------------------------------------

procedure departman_ekle is
    v_cur_id pls_integer;
    v_sql varchar2(100);
    v_rows integer;

    departid_array     DBMS_SQL.NUMBER_TABLE;
    deptname_array     DBMS_SQL.VARCHAR2_TABLE;
begin
    departid_array(1) := 116;
    departid_array(2) := 117;
    departid_array(3) := 118;

    deptname_array(1) := 'Uzay Bilimleri';
    deptname_array(2) := 'Yapay Zeka Ar-Ge';
    deptname_array(3) := 'Geri Dönüşüm Ar-Ge';

    v_sql := 'insert into departman values (:depid_array, :depname_array)';
    v_cur_id := DBMS_SQL.OPEN_CURSOR;
    DBMS_SQL.PARSE(v_cur_id, v_sql, DBMS_SQL.NATIVE);
    DBMS_SQL.BIND_ARRAY(v_cur_id, ':depid_array', departid_array);
    DBMS_SQL.BIND_ARRAY(v_cur_id, ':depname_array', deptname_array);
    v_rows := DBMS_SQL.EXECUTE(v_cur_id);
    DBMS_SQL.CLOSE_CURSOR(v_cur_id);
end;
```

```
-----------------------------------------
--DBMS_SQL Örnek-4
-----------------------------------------

 procedure personel_yazdir is
        v_cur_id pls_integer;
        v_sql varchar2(100);
        v_rows integer;

        col_ad  varchar2(30);
        col_maas number;
    begin

      v_sql := 'Select ad, maas from personel where unvan=''UZMAN''';
      v_cur_id := DBMS_SQL.OPEN_CURSOR;
      DBMS_SQL.PARSE(v_cur_id, v_sql, DBMS_SQL.NATIVE);
      v_rows := DBMS_SQL.EXECUTE(v_cur_id);

      DBMS_SQL.DEFINE_COLUMN(v_cur_id, 1, col_ad, 30);
      DBMS_SQL.DEFINE_COLUMN(v_cur_id, 2, col_maas);

      while DBMS_SQL.FETCH_ROWS(v_cur_id) > 0 loop

        DBMS_SQL.COLUMN_VALUE(v_cur_id, 1, col_ad);
        DBMS_SQL.COLUMN_VALUE(v_cur_id, 2, col_maas);
        DBMS_OUTPUT.PUT_LINE(col_ad || ' : ' || col_maas);

      end loop;
      DBMS_SQL.CLOSE_CURSOR(v_cur_id);
    end;
```

```
------------------------------------------------
--ALIŞTIRMALARIN CEVAPLARI
------------------------------------------------

-----------------------------------------
--EXECUTE IMMEDIATE - Sorgu
-----------------------------------------

declare
    v_sql varchar2(500) := 'select fruit_a from basket_a where id_a = :b1';
    v_fruit varchar2(25);

begin

    for i in 1..5 loop

        execute immediate v_sql into v_fruit using i;
        dbms_output.put_line(v_fruit);

    end loop;

end;

-----------------------------------------
--EXECUTE IMMEDIATE - DML
-----------------------------------------

declare
    type cars_type is table of varchar2(30);

    v_car_brand cars_type := cars_type('Bugatti', 'McLaren', 'Lamborghini');
    v_car_price cars_type := cars_type('200000', '250000', '300000');

    v_sql varchar2(100) := 'insert into cars(id, brand, price) values(:b1, :b2, :b3)';
    v_max_id pls_integer;

begin

    select max(id) into v_max_id from cars;

    for i in v_car_brand.first..v_car_brand.last loop

        execute immediate v_sql using v_max_id + i, v_car_brand(i), to_number(v_car_price(i));

    end loop;
end;


-----------------------------------------
--Data Dictionary ile Dinamik SQL Yazma
-----------------------------------------

SELECT
    'ALTER TABLE '||TABLE_NAME||
    ' DISABLE CONSTRAINT '||CONSTRAINT_NAME||';'
FROM USER_CONSTRAINTS;
```

```
-----------------------------------------
--BULK COLLECT INTO - Limit
-----------------------------------------

declare
    cursor crs_stock is
        select product_name, company_name,
            sum(units_in_stock) stock_amount
        from products p, suppliers s
        where p.supplier_id = s.supplier_id
        group by product_name, company_name
        order by 3 desc;

    type t_stock is table of crs_stock%rowtype;
    v_stock_info t_stock;

begin

    open crs_stock;

        fetch crs_stock
            bulk collect into v_stock_info limit 10;

        for i in 1 ..v_stock_info.count
        loop
            dbms_output.put_line('[Product Name:' || v_stock_info(i).product_name ||
                                 '] [Sup. Comp. Name:' || v_stock_info(i).company_name ||
                                 '] [Stock Amunt:' || v_stock_info(i).stock_amount ||']');
        end loop;

   close crs_stock;

end;
```

```
-----------------------------------------
--BULK COLLECT INTO - FORALL
-----------------------------------------

--alter table order_details add last_price number;
--alter table order_details add order_details_id number;
--update order_details set order_details_id=rownum;

declare
  cursor crs_order_details is
     select * from order_details;

    type t_order_details is table of crs_order_details%rowtype;
    v_order_details t_order_details;

    v_time_start number;
    v_time_end number;

begin

    v_time_start := DBMS_UTILITY.get_time;

    for order_detail_row in crs_order_details loop
        update order_details set last_price = round(unit_price*quantity*(100-discount)/100,2)
        where order_details_id = order_detail_row.order_details_id;
    end loop;

    v_time_end := DBMS_UTILITY.get_time;
    dbms_output.put_line('For loop inserts: ' || (v_time_end - v_time_start));

    v_time_start := DBMS_UTILITY.get_time;
    open crs_order_details;

        fetch crs_order_details
            bulk collect into v_order_details;

        forall i in v_order_details.first..v_order_details.last
            update order_details set last_price = round(unit_price*quantity*(100-
discount)/100,2)
            where order_details_id = v_order_details(i).order_details_id;

    close crs_order_details;
    v_time_end := DBMS_UTILITY.get_time;
    dbms_output.put_line('Forall inserts: ' || (v_time_end - v_time_start));

   commit;

end;
```

```
-----------------------------------------
--DBMS_SQL
-----------------------------------------

create or replace function create_email(p_student_name varchar2, p_course_name varchar2)
return varchar2
is
    v_return_value  varchar2(500);
begin
    select
        lower(replace(p_student_name,' ','.'))||'@'||
        lower(replace(p_course_name,' ','.'))||'.com'
        into v_return_value
    from dual;

    return v_return_value;
end;

create or replace procedure update_email(p_course_name varchar2)
is
    v_cur_id pls_integer;
    v_up_rows number;
    v_sql varchar2(100);

begin

    v_sql := 'update student set email = create_email(name, course_name) where course_name =
:cn';
    v_cur_id := DBMS_SQL.OPEN_CURSOR;
    DBMS_SQL.PARSE(v_cur_id, v_sql, DBMS_SQL.NATIVE);
    DBMS_SQL.BIND_VARIABLE(v_cur_id, ':cn', p_course_name);
    v_up_rows := DBMS_SQL.EXECUTE(v_cur_id);
    DBMS_SQL.CLOSE_CURSOR(v_cur_id);
    dbms_output.put_line('Updated rows:' || v_up_rows);
    commit;
end;
```