DB1 Syntax-Cheatsheet

Felix Pojtinger

June 29, 2021

Contents

DB1 Syntax-Cheatsheet

> "Come, let us go down and confuse their language so they will not understand each other" - Genesis 11:7, *Die Bibel*

Mehr Details unter https://github.com/pojntfx/uni-db1-notes. Dieses Dokument ist nur als Schnell-Übersicht gedacht.

Data Definition Language

Tabellen

Tabelle erstellen

```
create table persons (
    person_id number primary key not null ,
    first_name varchar2(50),
    last_name varchar2(50) default 'Duck' not null
);
```

Tabelle löschen

```
drop table persons;
```

Tabelle umbenennen

```
alter table persons rename to people;
```

Spalten

Spalten hinzufügen

```
alter table persons add ( phone varchar2(20), email varchar2(100) )
```

Spalten bearbeiten

```
alter table persons modify ( birthdate date null, email varchar2(255) );
```

Spalten löschen

```
alter table persons drop column birthdate;
```

Constraints

Constraints hinzufügen

```
alter table purchase_orders add constraint purchase_orders_order_id_pk primary key(order_id)
```

Constraints löschen

```
alter table purchase_orders drop constraint purchase_orders_order_id_pk;
```

Views

Views erstellen

```
create view employees_years_of_service
as select
    employee_id, first_name || ' ' || last_name as full_name,
    floor(months_between(current_date, hire_date) / 12) as years_of_service
from employees;
```

Views löschen

```
drop view employees_years_of_service;
```

Indizes

Indizes erstellen

```
create index members_full_name on members(first_name, last_name);
```

Indizes löschen

```
drop index members_full_name;
```

Trigger

Trigger erstellen

```
create trigger customers_credit_trigger
    before update of credit_limit
    on customers
declare
    current_day number;
begin
    current_day := extract(day from sysdate);

    if current_day between 28 and 31 then
        raise_application_error(-20100, 'Locked at the end of the month');
    end if;
end;
```

Trigger löschen

```
drop trigger customers_credit_trigger;
```

Exceptions handlen

```
create trigger users_ensure_trigger
    before update
    on users
    for each row
declare
    user_invalid exception;
    pragma exception_init(user_invalid, -20555);
begin
    raise user_invalid;

    exception
        when user_invalid then
            raise_application_error(-20555, 'User is invalid');
        when others then
            dbms_output.put_line('Unexpected error: ' || sqlerrm);
end;
```

Functions

Function erstellen

```
create or replace function get_my_sum( a integer, b integer ) return integer
is
        multiplier number := 2;
begin
```

```
        return a + b * multiplier;
end;
```

Function callen

```
select get_my_sum(1, 2) from dual;
```

Function löschen

```
drop function get_my_sum;
```

Procedure

Procedure erstellen

```
create or replace procedure get_sum ( a integer, b integer )
is
        multiplier number := 2;
        result number := 0;
begin
        result := a + b * multiplier;

        insert into results ( result ) values ( result );
end;
```

Procedure callen

```
exec get_sum(1, 2);
```

Procedure löschen

```
drop procedure get_sum;
```

Data Manipulation Language

Datentypen

- CHAR|CHARACTER (size)
- VARCHAR2 (size)
- DATE
- INTERVAL YEAR TO MONTH
- INTERVAL DAY TO SECOND
- INTEGER|INT
- NUMBER (precision [, scale])
- FLOAT (precision)

Zeilenoperationen

Insert

```
insert into discounts(
    discount_name,
    amount,
    start_date,
    expired_date
) values (
    'Summer Promotion',
    9.5,
    date '2017-05-01',
    date '2017-08-31'
)
```

Update

```
update products
set list_price = 420
where list_price < 69;
```

Delete

```
delete from products
where list_price > 69;
```

Unions

Gleiche Anzahl von Spalten, mehr Zeilen.

```
select
    first_name,
    last_name,
    email,
    'contact' as role
from contacts
union select
    first_name,
    last_name,
    email,
    'employee' as role
from employees order by role
```

Joins

Mehr Spalten & mehr Zeilen

Inner Join

```
select
    a.id as id_a,
    a.color as color_a,
    b.id as id_b,
    b.color as color_b
from palette_a a
inner join palette_b b using(color);
```

Left Outer Join

```
select
    a.id as id_a,
    a.color as color_a,
    b.id as id_b,
    b.color as color_b
from palette_a a
left outer join palette_b b using(color);
```

Right Outer Join

```
select
    a.id as id_a,
    a.color as color_a,
    b.id as id_b, b.color as color_b
from palette_a a
right outer join palette_b b using(color);
```

Full Outer Join

```
select
    a.id as id_a,
    a.color as color_a,
    b.id as id_b,
    b.color as color_b
from palette_a a
full outer join palette_b b using(color);
```

Trigger

Insert-Trigger `:old` ist nicht vorhanden.

```
create or replace trigger customers_credit_trigger
    before insert of credit_limit
    on customers
declare
    current_day number;
```

```
begin
    current_day := extract(day from sysdate);

    if current_day between 28 and 31 then
        raise_application_error(-20100, 'Locked at the end of the month');
    end if;
end;
```

Update-Trigger

```
create or replace trigger customers_credit_limit_trigger
    before update of credit_limit
    on customers
    for each row
    when (new.credit_limit > 0)
begin
    if :new.credit_limit >= 2*:old.credit_limit then
        raise_application_error(-20101, 'The new credit cannot be more than double the old c
    end if;
end;
```

Delete-Trigger

`:new` ist nicht vorhanden.

```
create or replace trigger customers_audit_trigger
    after delete
    on customers
    for each row
declare
    transaction_type varchar2(10);
begin
    transaction_type := case
        when updating then 'update'
        when deleting then 'delete'
    end;

    insert into audits(
        table_name,
        transaction_name,
        by_user,
        transaction_date
    ) values (
        'customers',
        transaction_type,
        user,
        sysdate
    );
```

```
end;
```

Instead-Of-Trigger

```
create or replace trigger create_customer_trigger
    instead of insert on customers_and_contacts
    for each row
declare
    current_customer_id number;
begin
    insert into customers(
        name,
        address,
        website,
        credit_limit
    ) values (
        :new.name,
        :new.address,
        :new.website,
        :new.credit_limit
    ) returning customer_id into current_customer_id;

    insert into contacts(
        first_name,
        last_name,
        email,
        phone,
        customer_id
    ) values (
        :new.first_name,
        :new.last_name,
        :new.email,
        :new.phone,
        current_customer_id
    );
end;
```

Ort der Verdammnis

menhir

Wenn einem der Syntax schon nicht kompliziert genug ist, dann darf man *vor* das `declare`-Statement eines Triggers auch noch folgendes sinnloses Konstrukt packen und statt `:new :neu` schreiben:

```
referencing new as neu old as alt
```

Danach hat man auch fünf Zeilen. Und fünf Hirnzellen weniger.

Wo wir schon dabei sind: Ist der sonst universelle Negations-Operator `!=` zu einfach? Zu simpel und zu verständlich? Wie wäre es mit `<>`; macht das genau selbe, ist aber komplizierter™!