

# IdeaBroker

**USER & INSTALLATION MANUAL**

**Bruno Madureira nr 2011161942**

**Fábio Silva nr 2010147721**

## Scripts:

Para a criação da Base de dados

```
CREATE TABLE login(name VARCHAR(50), password VARCHAR(50),isroot NUMERIC(1), PRIMARY KEY(name))
```

```
CREATE TABLE saldos(name VARCHAR(50), creditos NUMERIC(20,5), PRIMARY KEY(name))
```

```
CREATE TABLE idea_table(acções_totais Numeric(30,0), tipo Numeric(2,0),texto VARCHAR(500), idea_key NUMERIC(15,0),file_name VARCHAR(100),file Blob ,PRIMARY KEY (idea_key,texto))
```

```
CREATE TABLE topicos (hashtag VARCHAR(500),topic_key NUMERIC(15,0),PRIMARY KEY(topic_key))
```

```
CREATE TABLE i_t (idea_key NUMERIC(15,0),topic_key NUMERIC(15,0), PRIMARY KEY(idea_key, topic_key))
```

```
CREATE TABLE shares_table (nome Varchar(500),acções Numeric (30,0),idea_key NUMERIC(15,0), preco NUMERIC(30,5),Primary key(nome,idea_key))
```

```
CREATE TABLE historico(vendedor VARCHAR(500),comprador VARCHAR(500),acções numeric(30,0),preco Numeric(30,5),idea_key NUMERIC(15,0),n_compra NUMERIC(15,0), PRIMARY KEY( vendedor,comprador,idea_key,n_compra) )
```

```
CREATE TABLE watchlist(name VARCHAR(50),idea_key NUMERIC(15,0),PRIMARY KEY(name))
```

```
CREATE TABLE walloffame(idea_key NUMERIC(15,0), texto VARCHAR(500),n_compra NUMERIC(15,0), PRIMARY KEY idea_key)
```

```
CREATE TABLE PEDIDOS ( NAME VARCHAR2 (50 ) , IDEA_KEY NUMBER (15) , LIMITE NUMBER (30,10) , NUMERO_ACÇÕES NUMBER (30) , ORDEM NUMBER (15) ,PRIMARY KEY( NAME, IDEA_KEY))
```

```
CREATE TABLE HALLOFFAME ( IDEA_KEY NUMBER (15), PRIMARY KEY ( IDEA_KEY ) )
```

## Sequences

<code>CREATE SEQUENCE IDEA_KEY_SEQUENCE</code>	<code>CREATE SEQUENCE TOPIC_KEY_SEQUENCE</code>
<code>MINVALUE 1</code>	<code>MINVALUE 1</code>
<code>MAXVALUE 9999999999999999999999999999999</code>	<code>MAXVALUE 9999999999999999999999999999999</code>
<code>INCREMENT BY 1</code>	<code>INCREMENT BY 1</code>
<code>START WITH 1</code>	<code>START WITH 1</code>
<code>NOCACHE</code>	<code>NOCACHE</code>
<code>NOORDER</code>	<code>NOORDER</code>
<code>NOCYCLE ;</code>	<code>NOCYCLE ;</code>

Estas sequencias são necessárias para que quando adicionamos um novo tópico ou uma nova ideia os identificadores únicos das tabelas vão ser incrementados em um valor para terem uma chave única

## Triggers

<p>create or replace</p> <pre> TRIGGER idea_key_trigger BEFORE INSERT ON idea_table FOR EACH ROW BEGIN :NEW.idea_key:=IDEA_KEY_SEQUENCE.NEXTVAL; END;</pre>	<p>create or replace</p> <pre> TRIGGER Topic_key_trigger BEFORE INSERT ON topicos FOR EACH ROW BEGIN :NEW.topic_key:=topic_Key_sequence.NEXTVAL; END;</pre>
---	---

Os triggers apresentados são os responsáveis por chamar as sequences momentos antes de adicionar tanto os tópicos como as ideias.

## Procedure

```
create or replace
PROCEDURE inserir(nome varchar2,login varchar2, root
number)
is
BEGIN
insert into Login values (nome, login,root);
END;
```

Este procedure serve para quando for adicionado um novo utilizador criar um novo elemento na tabela Login.

### Functions

**create or replace**

```
FUNCTION adiciona_dinheiro (user IN VARCHAR2,quantidade IN NUMERIC)
    RETURN NUMBER IS Retcreditos NUMBER(20,5);
PRAGMA AUTONOMOUS_TRANSACTION;
BEGIN
    Retcreditos:=dinheiro_actual(user)+quantidade;
    UPDATE saldos SET creditos=retcreditos where name=user;
    COMMIT;
    RETURN Retcreditos;

    END adiciona_dinheiro;
```

Esta função é usada para as transações ou seja quando é efectuada uma nova transação de uma ideia esta função vai fazer o “pagamento” ao vendedor .

**create or replace**

```
FUNCTION adicionarlogin (nome varchar2,login varchar2, root number)
    RETURN NUMBER IS ret number(20,5) ;
PRAGMA AUTONOMOUS_TRANSACTION;
BEGIN
    insert into Login values (nome, login,root);
    select isRoot into ret from login where name=nome ;
    COMMIT;
    return ret;

    END adicionarlogin;
```

create or replace

```
FUNCTION retira_dinheiro (user IN VARCHAR2,quantidade IN NUMERIC)
    RETURN NUMBER IS Retcreditos NUMBER(20,5);
PRAGMA AUTONOMOUS_TRANSACTION;
BEGIN

Retcreditos:=dinheiro_actual(user)-quantidade;
        UPDATE saldos SET creditos=retcreditos where name=user;
        COMMIT;
RETURN Retcreditos;

END retira_dinheiro;
```

Esta função é usado para as transações ou seja quando e efectuado uma nova transação de uma ideia esta função vai retirar o dinheiro ao comprador para adicionar ao vendedor.

create or replace

```
FUNCTION dinheiro_actual (user IN VARCHAR2) RETURN NUMBER IS
Retcreditos saldos.creditos%type;

BEGIN

        SELECT creditos INTO Retcreditos FROM saldos where name=user;
        RETURN Retcreditos;

END dinheiro_actual;
```

Esta função vai ser usada pelas funções `retira_dinheiro` e `adiciona_dinheiro` para ser feito o update ao valor de créditos que o vendedor e o comprador vão ter depois de efectuado a transação.