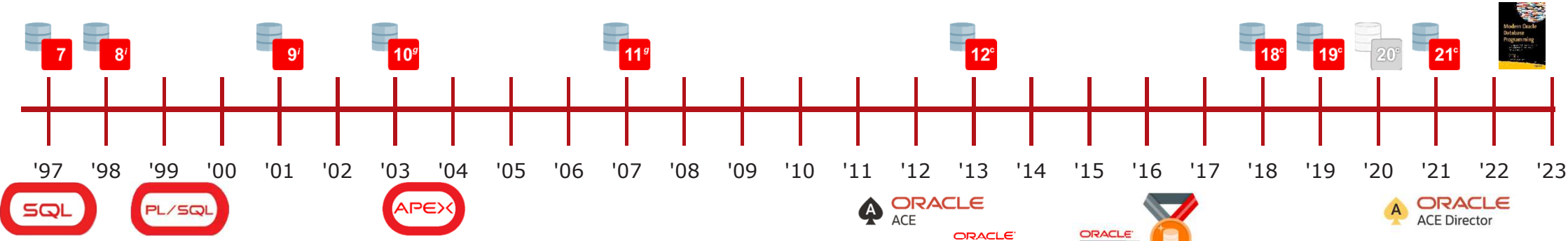
QUALOGY

PBarel@Qualogy.com
http://blog.bar-solutions.com

QUALOGY

About me…



bar-solutions.com blog http://blog.bar-solutions.com

All things ORACLE http://allthingsoracle.com

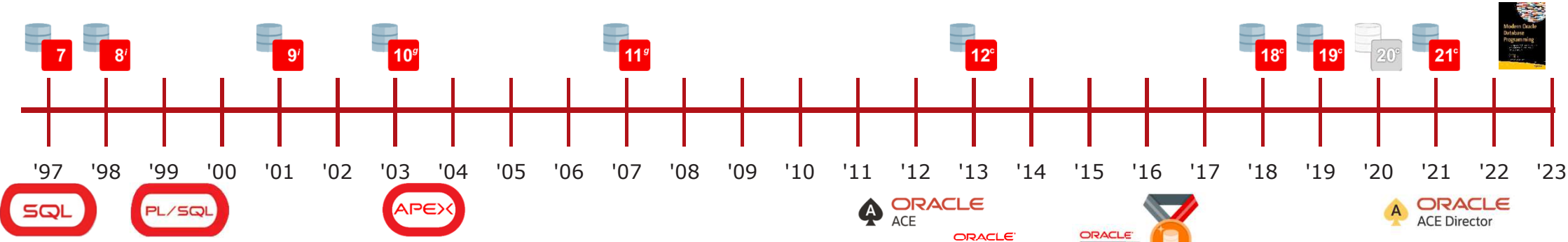
OTECH MAGAZINE http://www.otechmag.com

www.red-gate.com/simple-talk/author/patrick-barel/

bar-solutions.com/otechmagazine.php

Plugins for PL/SQL Developer
http://plugins.bar-solutions.com

About me…



bar-solutions.com blog http://blog.bar-solutions.com

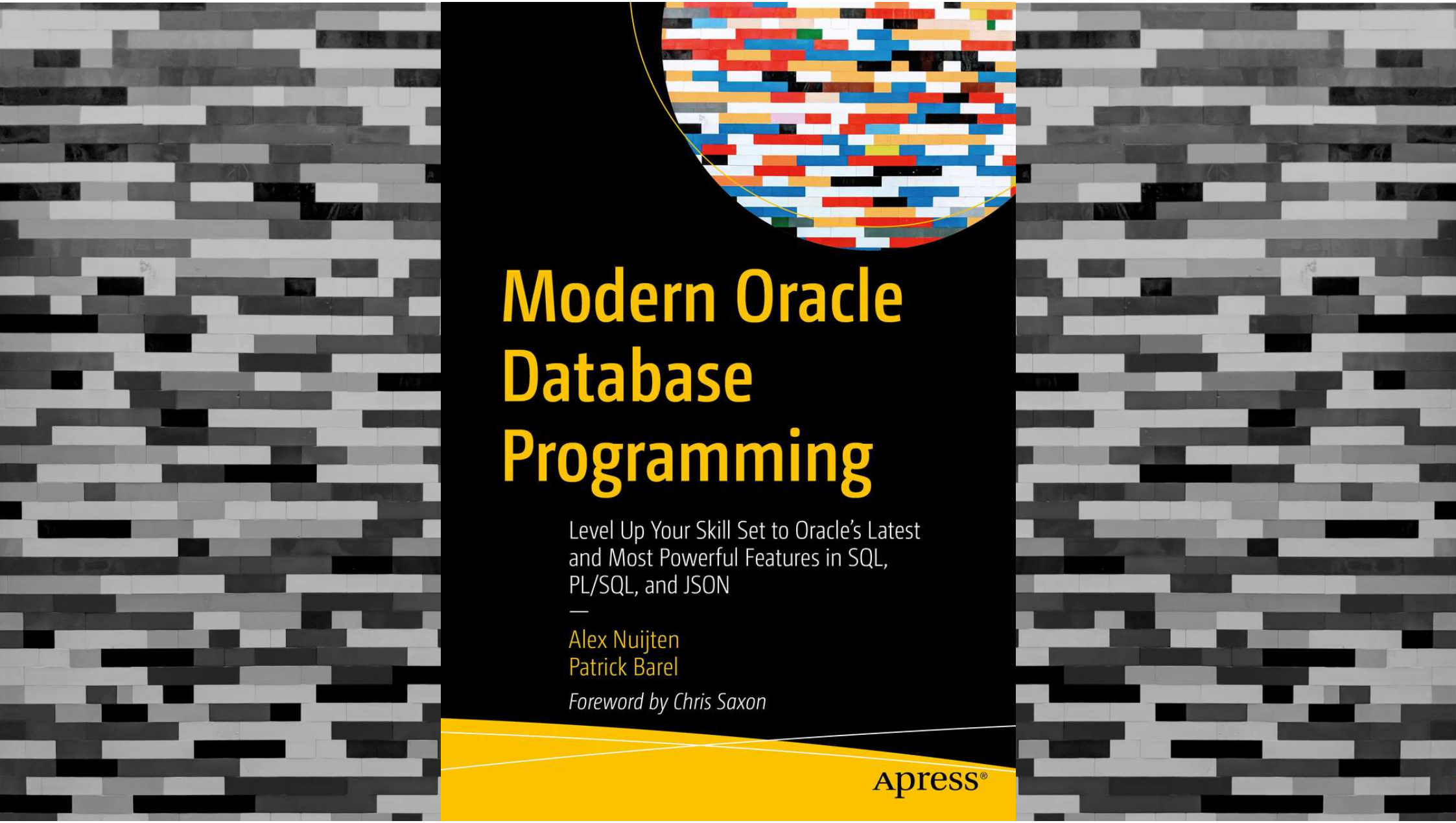
All things ORACLE http://allthingsoracle.com

OTECH MAGAZINE http://www.otechmag.com

www.red-gate.com/simple-talk/author/patrick-barel/

bar-solutions.com/otechmagazine.php

Plugins for PL/SQL Developer
http://plugins.bar-solutions.com


Modern Oracle Database Programming
Level Up Your Skill Set to Oracle's Latest and Most Powerful Features in SQL, PL/SQL, and JSON
—
Alex Nuijten
Patrick Barel
Foreword by Chris Saxon
Apress®

Contact me…

@patch72

PBarel@Qualogy.com

Patrick.Barel@GMail.com

patrick@bar-solutions.com

Patrick.Barel@GMail.com

Patrick Barel

Oracle ACE Program

500+ technical experts helping peers globally

The **Oracle ACE Program** recognizes and rewards community members for their technical and community contributions to the Oracle community

3 membership tiers

Oracle ACE Director | Oracle ACE Pro | Oracle ACE Associate

Oracle ACE

Nominate

~~yourself~~ or someone you know:

ace.oracle.com/nominate

For more details on Oracle ACE Program:
ace.oracle.com

Connect: **aceprogram_ww@oracle.com** Facebook.com/OracleACEs @oracleace

Oracle Cloud Infrastructure

New Free Tier

oracle.com/cloud/free

Always Free

Services you can use for unlimited time

+

30-Day Free Trial

Free credits you can use for more services


MASH
PROGRAM
Mentor and Speaker Hub
Our goal is to *connect* speakers with mentors to assist in *preparing* technical sessions and *improving* presentation skills
Interested? Read more and get in touch
https://mashprogram.wordpress.com

SYMPOSIUM 42

Created by the community, to support the community

Sharing of reliable knowledge
Supporting the various user groups and individuals

@sym_42

https://sym42.org/

Get Your Money's Worth Out Of Your Database

Patrick Barel, Qualogy October 18, 2023

QUALOGY

It doesn’t matter how many resources you have.

If you don’t know how to use them,
it will never be enough

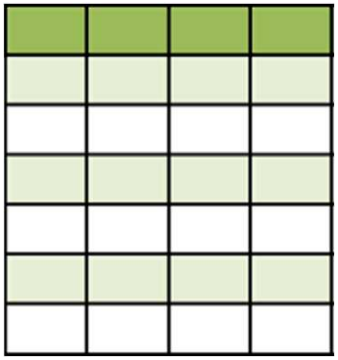
Database

Data integrity/quality

Data integrity/quality

```
create table demo.emp_unprotected
(
  empno  number(4) not null
, ename  varchar2(10)
, sal    number(7,2)
, deptno number(2)
)
```

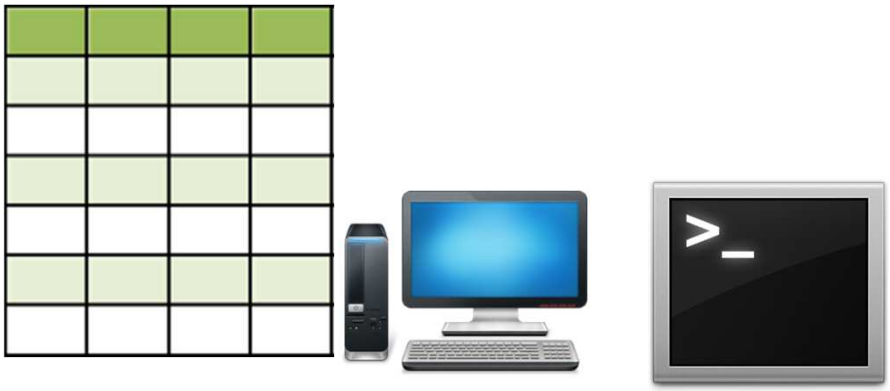
Data integrity/quality



```
insert into demo.emp_unprotected values (7369, 'SMITH',  800,  20);
insert into demo.emp_unprotected values (7499, 'ALLEN',  1600, 30);
insert into demo.emp_unprotected values (7521, 'WARD',   1250, 30);
insert into demo.emp_unprotected values (7566, 'JONES',  2975, 20);
…
insert into demo.emp_unprotected values (7900, 'JAMES',  950,  30);
insert into demo.emp_unprotected values (7902, 'FORD',   3000, 20);
insert into demo.emp_unprotected values (7934, 'MILLER', 1300, 10);
```

Data integrity/quality

- Salary not over 7500
- No duplicate employee numbers
- Existing department



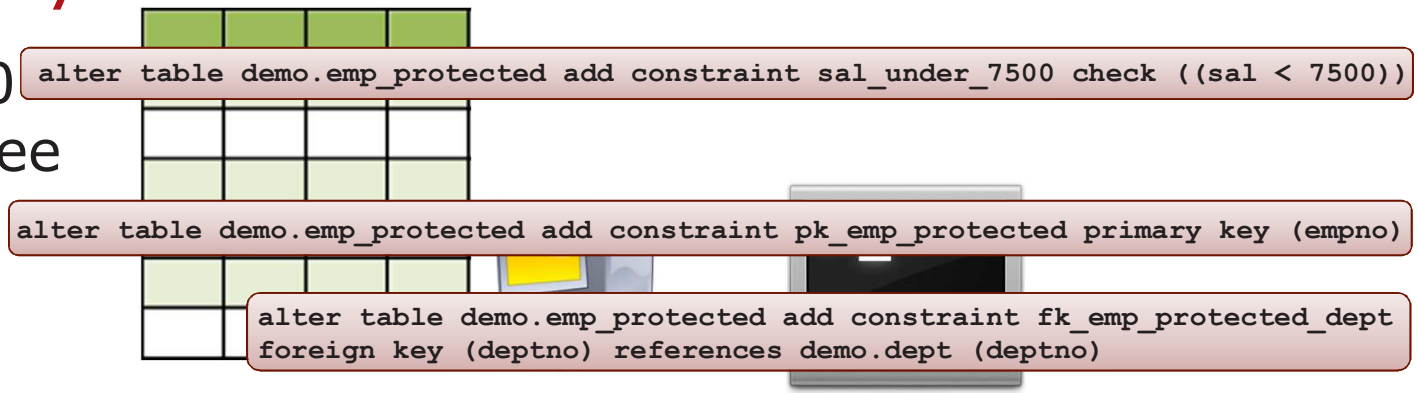
```
update demo.emp_unprotected e
   set e.sal = 2 * e.sal
```

```
insert into demo.emp_unprotected
            (empno, ename, sal, deptno)
     values (7900, 'BAREL', 1000, 10)
```

```
update demo.emp_unprotected e
   set e.deptno = 50
 where e.ename = 'SCOTT'
```

Data integrity/quality

- Salary not over 7500
- No duplicate employee numbers
- Existing department



```
alter table demo.emp_protected add constraint sal_under_7500 check ((sal < 7500))
```

```
alter table demo.emp_protected add constraint pk_emp_protected primary key (empno)
```

```
alter table demo.emp_protected add constraint fk_emp_protected_dept
foreign key (deptno) references demo.dept (deptno)
```

```
update demo.emp_protected e
   set e.sal = 2 * e.sal
```

```
insert into demo.emp_protected
            (empno, ename, sal, deptno)
     values (7900, 'BAREL', 1000, 10)
```

```
update demo.emp_protected e
   set e.deptno = 50
 where e.ename = 'SCOTT'
```

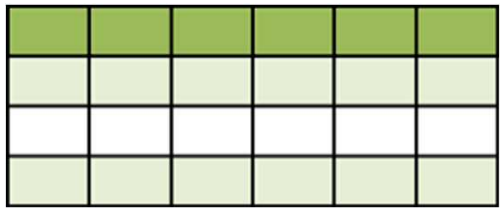
dataintegrity.sql

Flashback queries

Flashback queries

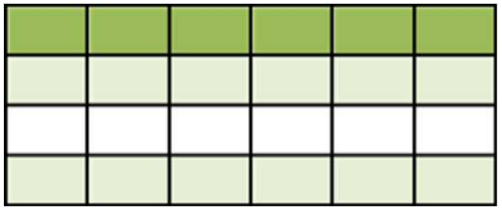
```
create table emp_fb
(
  empno  NUMBER(4) not null,
  ename  VARCHAR2(10),
  sal    NUMBER(7,2),
  deptno NUMBER(2)
)
```

Flashback queries



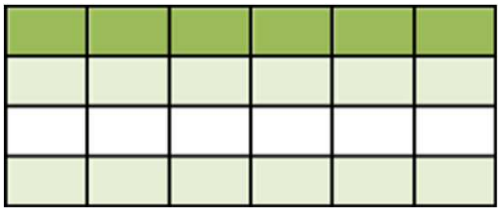
```
…
insert into emp_fb values (7782, 'CLARK',  2450, 10);
insert into emp_fb values (7788, 'SCOTT',  3000, 20);
insert into emp_fb values (7839, 'KING',   5000, 10);
insert into emp_fb values (7844, 'TURNER', 1500, 30);
insert into emp_fb values (7876, 'ADAMS',  1100, 20);
insert into emp_fb values (7900, 'JAMES',  950,  30);
insert into emp_fb values (7902, 'FORD',   3000, 20);
insert into emp_fb values (7934, 'MILLER', 1300, 10);
```

Flashback queries



```
EMPNO ENAME                SAL
----- ---------- ---------
 7782 CLARK           2650.00
 7839 KING            5200.00
 7934 MILLER          1500.00

select e.empno, e.ename, e.sal
  from emp_fb e
 where e.deptno = 10
```

Flashback queries



```
EMPNO ENAME                SAL
----- ---------- ---------
 7782 CLARK           2650.00
 7839 KING            5200.00
 7934 MILLER          1500.00
```

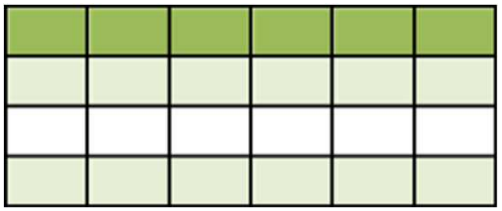
```
select e.empno, e.ename, e.sal
  from emp_fb e
 where e.deptno = 10
```

```
update emp_fb e
   set e.sal = e.sal + 200
 where e.deptno = 10
```

```
EMPNO ENAME                SAL
----- ---------- ---------
 7782 CLARK           2650.00
 7839 KING            5200.00
 7934 MILLER          1500.00
```

```
select e.empno, e.ename, e.sal
  from emp_fb e
 where e.deptno = 10
```

Flashback queries



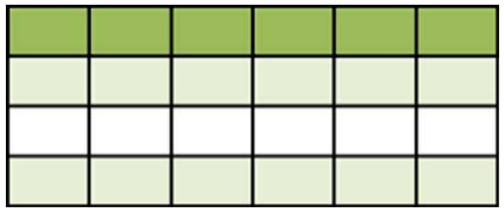
```
EMPNO ENAME                SAL
----- ---------- ---------
 7782 CLARK           2850.00
 7839 KING            5400.00
 7934 MILLER          1700.00
```

```
select e.empno, e.ename, e.sal
  from emp_fb e
 where e.deptno = 10

update emp_fb e
   set e.sal = e.sal + 200
 where e.deptno = 10

commit
```

```
EMPNO ENAME                SAL
----- ---------- ---------
 7782 CLARK           2650.00
 7839 KING            5200.00
 7934 MILLER          1500.00
```

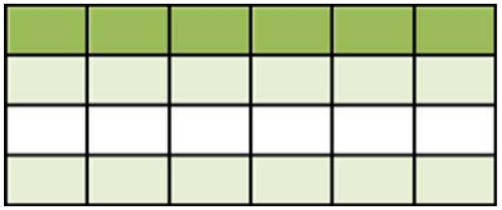
```
select e.empno, e.ename, e.sal
  from emp_fb e
 where e.deptno = 10
```

Flashback queries



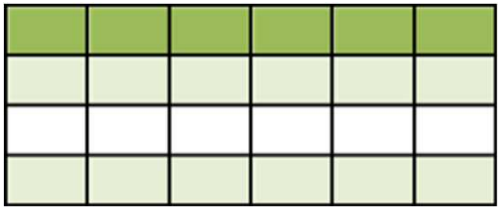
```
truncate table emp_fb

…
insert into emp_fb values (7782, 'CLARK',  2450, 10);
insert into emp_fb values (7788, 'SCOTT',  3000, 20);
insert into emp_fb values (7839, 'KING',   5000, 10);
insert into emp_fb values (7844, 'TURNER', 1500, 30);
insert into emp_fb values (7876, 'ADAMS',  1100, 20);
insert into emp_fb values (7900, 'JAMES',  950,  30);
insert into emp_fb values (7902, 'FORD',   3000, 20);
insert into emp_fb values (7934, 'MILLER', 1300, 10);
```

Flashback queries



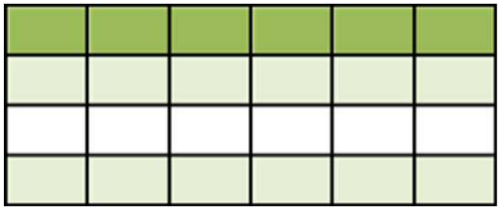
```
EMPNO ENAME                SAL
----- ---------- ---------
 7782 CLARK           2650.00
 7839 KING            5200.00
 7934 MILLER          1500.00

select e.empno, e.ename, e.sal
  from emp_fb e
 where e.deptno = 10
```

Flashback queries



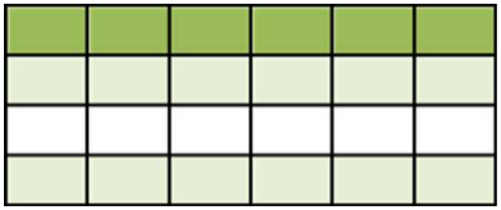
```
EMPNO ENAME                SAL
----- ---------- ---------
 7782 CLARK           2650.00
 7839 KING            5200.00
 7934 MILLER          1500.00

select e.empno, e.ename, e.sal
  from emp_fb e
 where e.deptno = 10

update emp_fb e
   set e.sal = e.sal + 200
 where e.deptno = 10
```

Flashback queries



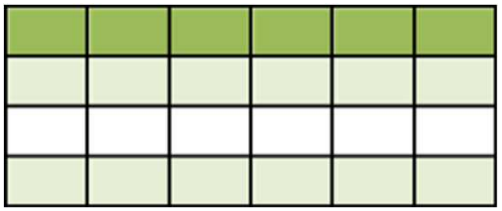
```
EMPNO ENAME                SAL
----- ---------- ---------
 7782 CLARK           2850.00
 7839 KING            5400.00
 7934 MILLER          1700.00

select e.empno, e.ename, e.sal
  from emp_fb e
 where e.deptno = 10

update emp_fb e
   set e.sal = e.sal + 200
 where e.deptno = 10

commit
```

Flashback queries



```
EMPNO ENAME                SAL
----- ---------- ---------
 7782 CLARK           2650.00
 7839 KING            5200.00
 7934 MILLER          1300.00

select e.empno, e.ename, e.sal
  from demo.emp_fb
       as of timestamp to_timestamp(sysdate - 1/(24*60*60)) e
 where e.deptno = 10
```

Flashback queries



```
EMPNO ENAME             SAL
----- ---------- ---------
 7782 CLARK         2650.00
 7839 KING          5200.00
 7934 MILLER        1500.00

select e.empno, e.ename, e.sal
  from demo.emp_fb
       as of timestamp to_timestamp(sys                t60)) e
 where e.deptno = 10

select e.empno, e.ename, e.sal
  from demo.emp_fb
       as of scn 23352983 e
 where e.deptno = 10
```

select dbms_flashback.
get_system_change_number
from dual

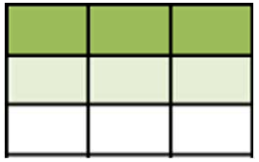
flashback_fb1.sql
flashback_fb2.sql
flashback_fb3.sql

Temporal validity

Temporal validity

```
create table demo.employees
( id number
, name varchar2(30)
, address varchar2(30)
)
```

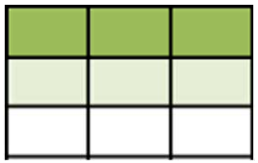
Temporal validity

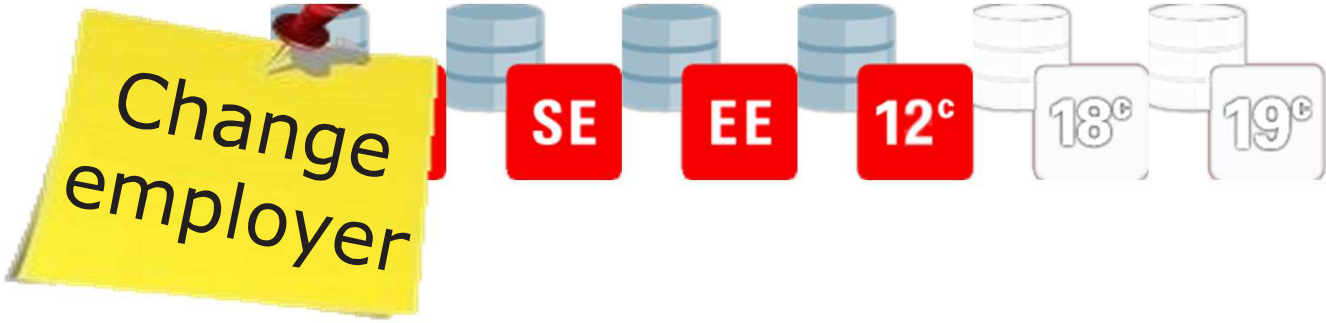


```
 ID NAME                      ADDRESS
--- ------------------------- -------------------------
  1 Patrick                   Edisonbaan 15
  2 Steven                    57 Chicago Boulevard
```

```
insert into demo.employees (id, name, address) values
                           (1, 'Patrick', 'Edisonbaan 15')
insert into demo.employees (id, name, address) values
                           (2, 'Steven', '57 Chicago Boulevard')

select * from demo.employees c
 order by c.id
```

Temporal validity



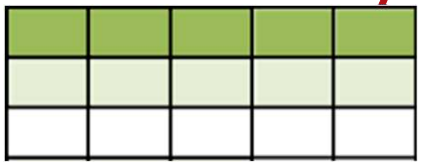



```
 ID NAME                      ADDRESS
--- ------------------------- -------------------------
  1 Patrick                   De Bruyn Kopsstraat 9
  2 Steven                    57 Chicago Boulevard
```

```
update demo.employees c
   set c.address = 'De Bruyn Kopsstraat 9'
 where c.id = 1
```

```
select * from demo.employees c
 order by c.id
```

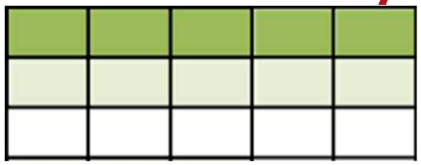
Temporal validity



```
 ID NAME                          ADDRESS
--- ---------------------------- --------------------------
  1 Patrick                      Edisonbaan 15
  2 Steven                       57 Chicago Boulevard
```

```
truncate table demo.employees
insert into demo.employees (id, name, address) values
                           (1, 'Patrick', 'Edisonbaan 15')
insert into demo.employees (id, name, address) values
                           (2, 'Steven', '57 Chicago Boulevard')
select * from demo.employees c
 order by c.id
alter table demo.employees add period for valid_time
```

Temporal validity



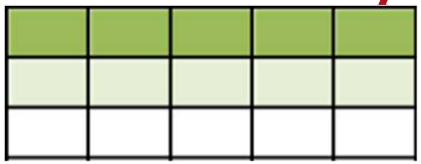

```
 ID NAME                      ADDRESS
--- ------------------------ ---------
  1 Patrick                  Edisonbaa
  2 Steven                   57 Chica
```

Columns are not visible when the table is described

```
desc demo.employees
Name      Type          Nullable
--------- ------------- --------
ID        NUMBER        Y
NAME      VARCHAR2(30)  Y
ADDRESS   VARCHAR2(30)  Y
```

```
truncate table demo.employees
insert into demo.employees (id,
                            (1,
insert into demo.employees (id,
                            (2, 'Steven
select * from demo.employees c
 order by c.id
alter table demo.employees add period for valid_time
```

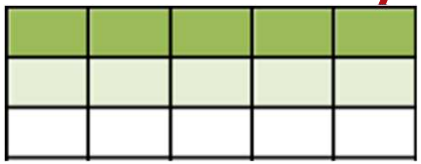
Temporal validity



```
 ID NAME                      ADDRESS                   VALID_FROM VALID_TILL
--- ------------------------- ------------------------- ---------- ----------
  1 Patrick                   Edisonbaan 15
  2 Steven                    57 Chicago Boulevard
```

```
select c.id
     , c.name
     , c.address
     , to_char(c.valid_time_start, 'MM-DD-YYYY') valid_from
     , to_char(c.valid_time_end,   'MM-DD-YYYY') valid_till
  from demo.employees c
 order by c.id
```

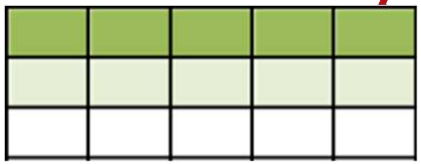
Temporal validity



```
 ID NAME                      ADDRESS                   VALID_FROM VALID_TILL
--- ------------------------- ------------------------- ---------- ----------
  1 Patrick                   Edisonbaan 15                        01-31-2018
  2 Steven                    57 Chicago Boulevard
```

```
select c.id
     , c.name
     , c.address
     , to_char(c.valid_time_start, 'MM-DD-YYYY') valid_from
     , to_char(c.valid_time_end,   'MM-DD-YYYY') valid_till
  from demo.employees c
 order by c.id
update demo.employees c
   set c.valid_time_end = to_date('01312018 235959', 'MMDDYYYY HH24MISS')
 where c.id = 1
```

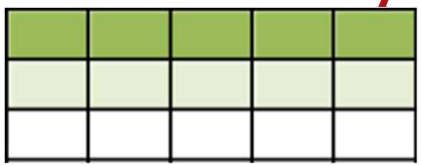
Temporal validity

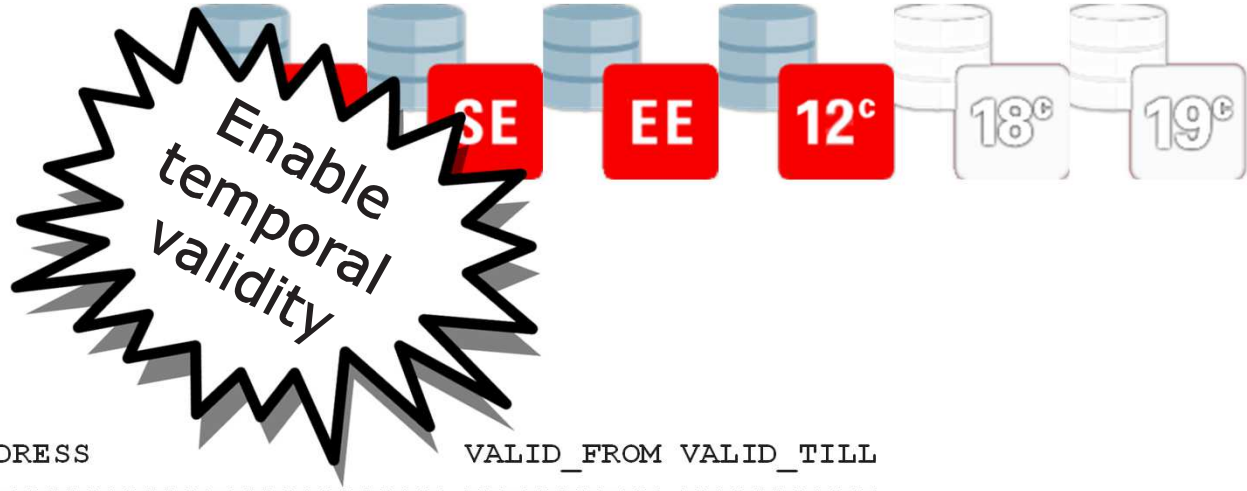


```
 ID NAME                      ADDRESS                   VALID_FROM VALID_TILL
--- ------------------------- ------------------------- ---------- ----------
  1 Patrick                   Edisonbaan 15                        01-31-2018
  2 Steven                    57 Chicago Boulevard      02-01-2018
  2 Steven                    57 Chicago Boulevard
```

```
select c.id
     , c.name
     , c.address
     , to_char(c.valid_time_start, 'MM-DD-YYYY') valid_from
     , to_char(c.valid_time_end,   'MM-DD-YYYY') valid_till
  from demo.employees c
 order by c.id
update demo.employees c
   set c.valid_time_end = to_date('01312018 235959', 'MMDDYYYY HH24MISS')
 where c.id = 1
insert into demo.employees(id, name, address, valid_time_start)
values (1,'Patrick','De Bruyn Kopsstraat
9',to_date('02012018','MMDDYYYY'))
```

Temporal validity

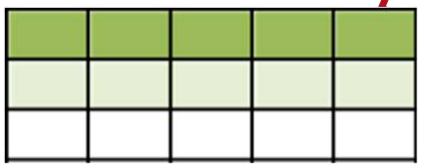


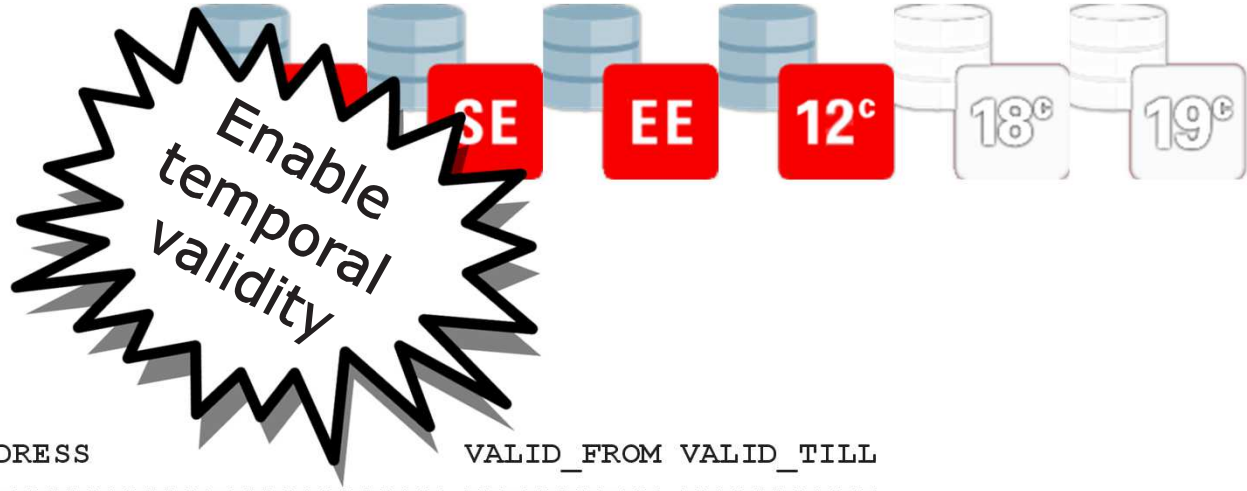



```
 ID NAME                      ADDRESS                   VALID_FROM VALID_TILL
--- ------------------------- ------------------------- ---------- ----------
  1 Patrick                   De Bruyn Kopsstraat 9     02-01-2018
  2 Steven                    57 Chicago Boulevard
```

```
select c.id
     , c.name
     , c.address
     , to_char(c.valid_time_start, 'MM-DD-YYYY') valid_from
     , to_char(c.valid_time_end,   'MM-DD-YYYY') valid_till
  from demo.employees c
 order by c.id
exec dbms_flashback_archive.enable_at_valid_time('CURRENT')
```

Temporal validity

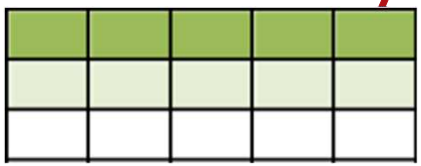


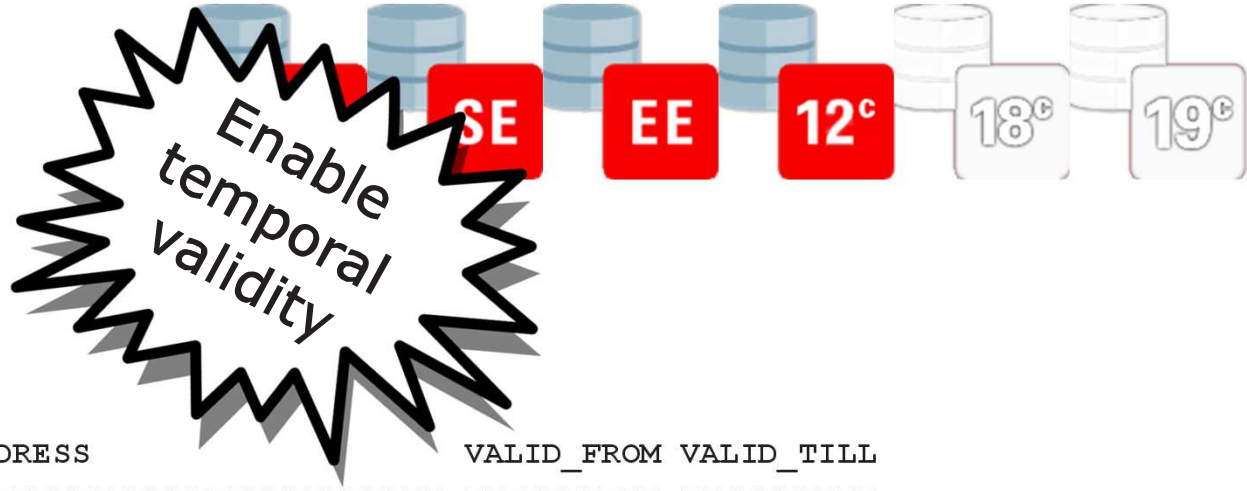



```
 ID NAME                      ADDRESS                   VALID_FROM VALID_TILL
--- ------------------------- ------------------------- ---------- ----------
  1 Patrick                   Edisonbaan 15             02-01-2018 01-31-2018
  2 Steven                    57 Chicago Boulevard
```

```
select c.id
     , c.name
     , c.address
     , to_char(c.valid_time_start, 'MM-DD-YYYY') valid_from
     , to_char(c.valid_time_end,   'MM-DD-YYYY') valid_till
  from demo.employees c
 order by c.id
exec dbms_flashback_archive.enable_at_valid_time('CURRENT')
exec dbms_flashback_archive.enable_at_valid_time('ASOF'
                                    , to_date('01312018','MMDDYYYY'))
```

Temporal validity

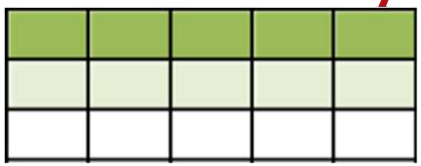


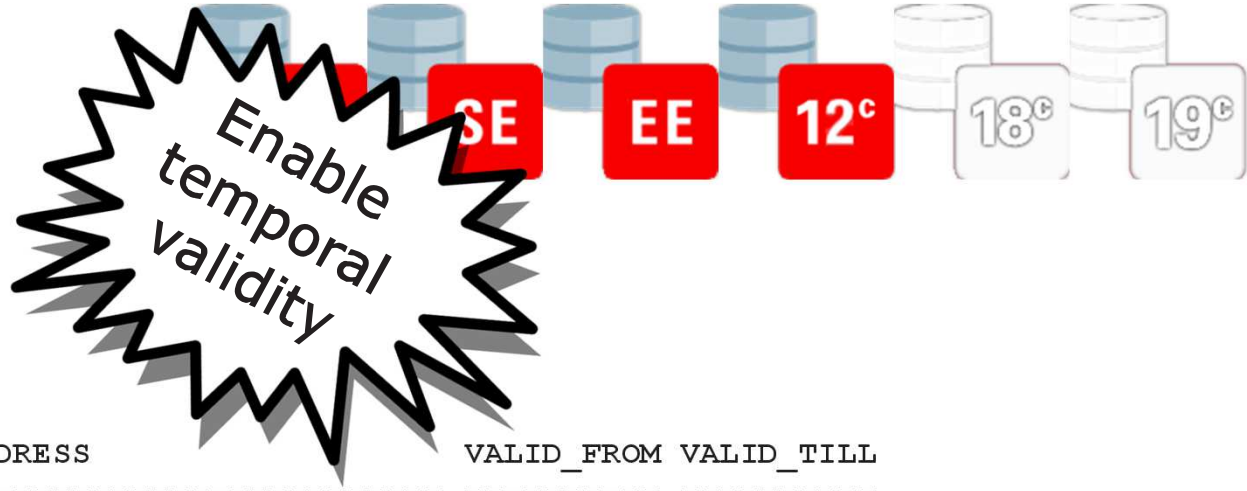



```
 ID NAME                      ADDRESS                   VALID_FROM VALID_TILL
--- ------------------------- ------------------------- ---------- ----------
  1 Patrick                   Edisonbaan 15             02-01-2018 01-31-2018
  2 Steven                    57 Chicago Boulevard
```

```
select c.id
     , c.name
     , c.address
     , to_char(c.valid_time_start, 'MM-DD-YYYY') valid_from
     , to_char(c.valid_time_end,   'MM-DD-YYYY') valid_till
  from demo.employees c
 order by c.id
exec dbms_flashback_archive.enable_at_valid_time('CURRENT')
exec dbms_flashback_archive.enable_at_valid_time('ASOF'
                                    , to_date('01312018','MMDDYYYY'))
exec dbms_flashback_archive.enable_at_valid_time('ASOF'
                                    , to_date('02012018','MMDDYYYY'))
```

Temporal validity






```
 ID NAME                      ADDRESS                   VALID_FROM VALID_TILL
--- ------------------------- ------------------------- ---------- ----------
  1 Patrick                   Edisonbaan 15             02-01-2018 01-31-2018
  1 Patrick                   Edisonbaan 9              02-01-2018
  2 Steven                    57 Chicago Boulevard
```

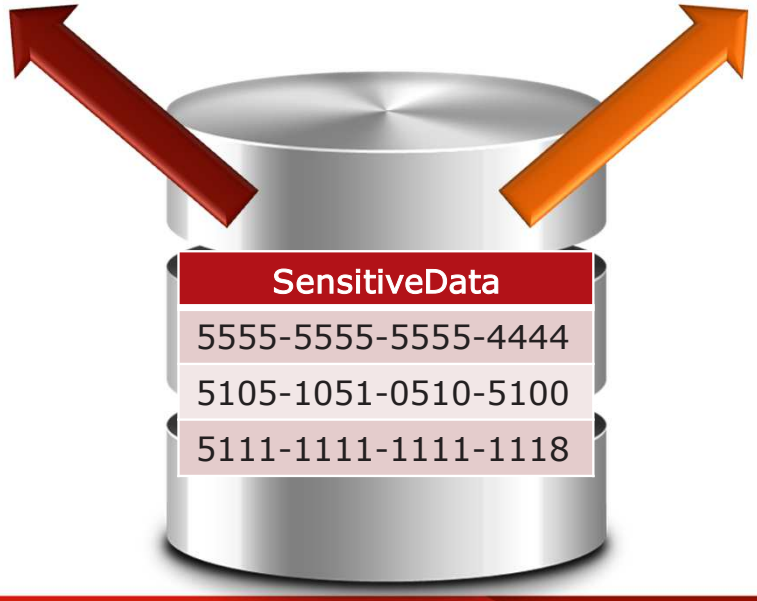
```
select c.id
     , c.name
     , c.address
     , to_char(c.valid_time_start, 'MM-DD-YYYY') valid_from
     , to_char(c.valid_time_end,   'MM-DD-YYYY') valid_till
  from demo.employees c
 order by c.id
exec dbms_flashback_archive.enable_at_valid_time('CURRENT')
exec dbms_flashback_archive.enable_at_valid_time('ASOF'
                                    , to_date('01312018','MMDDYYYY'))
exec dbms_flashback_archive.enable_at_valid_time('ASOF'
                                    , to_date('02012018','MMDDYYYY'))
exec dbms_flashback_archive.enable_at_valid_time('ALL')
```

temporal_validity.sql

Data security

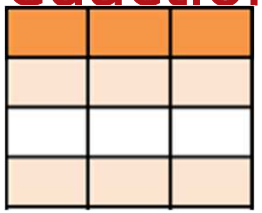
Data redaction

SensitiveData (Clear)	SensitiveData (Redacted)
5555-5555-5555-4444	XXXX-XXXX-XXXX-4444
5105-1051-0510-5100	XXXX-XXXX-XXXX-5100
5111-1111-1111-1118	XXXX-XXXX-XXXX-1118



SensitiveData
5555-5555-5555-4444
5105-1051-0510-5100
5111-1111-1111-1118

Data redaction

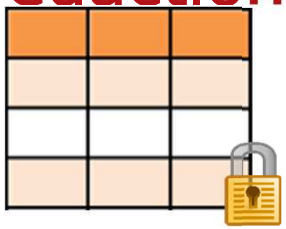


```
create table SensitiveData (contract varchar2(23))

insert into SensitiveData(contract) values ('5555-5555-5555-4444');
insert into SensitiveData(contract) values ('5105-1051-0510-5100');
insert into SensitiveData(contract) values ('5111-1111-1111-1118');
```

Data redaction

```
begin
  dbms_redact.add_policy(
    object_schema       => 'DEMO'
   ,object_name         => 'SENSITIVEDATA'
   ,policy_name         => 'PARTIALMASKDEMO'
   ,column_name         => 'CONTRACT'
   ,function_type       => DBMS_REDACT.PARTIAL
   ,function_parameters =>
      'VVVVFVVVVFVVVVFVVVV,VVVV-VVVV-VVVV-VVVV,X,1,12'
   ,expression          =>
     q'[SYS_CONTEXT ('USERENV', 'SESSION_USER') <> 'BU']');
end;
```

Data redaction



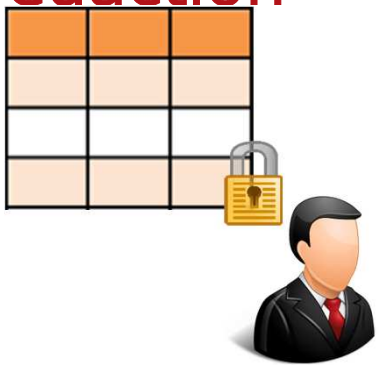
Different redaction types available

DBMS_REDACT.NONE
DBMS_REDACT.FULL
DBMS_REDACT.PARTIAL
DBMS_REDACT.RANDOM

http://docs.oracle.com/database/121/ARPLS/d_redact.htm

```
begin
  dbms_redact.add_policy(
    object_schema       => 'DEMO'
   ,object_name         => 'SENSITIVEDATA'
   ,policy_name         => 'PARTIALMASKDEMO'
   ,column_name         => 'CONTRACT'
   ,function_type       => DBMS_REDACT.PARTIAL
   ,function_parameters =>
      'VVVVFVVVVFVVVVFVVVV,VVVV-VVVV-VVVV-VVVV,X,1,12'
   ,expression          =>
     q'[SYS_CONTEXT ('USERENV', 'SESSION_USER') <> 'BU']');
end;
```

Data redaction



```
select * from demo.SensitiveData
```

```
CONTRACT
-----------------------
5555-5555-5555-4444
5105-1051-0510-5100
5111-1111-1111-1118
```

```
select * from demo.SensitiveData
```

```
CONTRACT
-----------------------
XXXX-XXXX-XXXX-4444
XXXX-XXXX-XXXX-5100
XXXX-XXXX-XXXX-1118
```

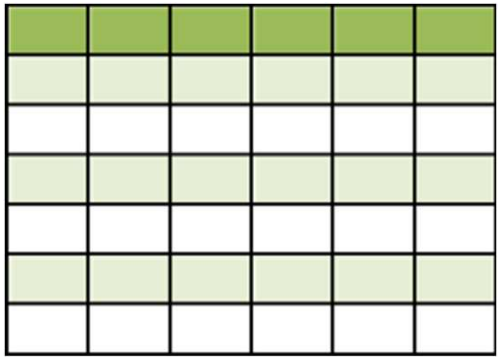
redaction.sql

Virtual private database

Virtual private database

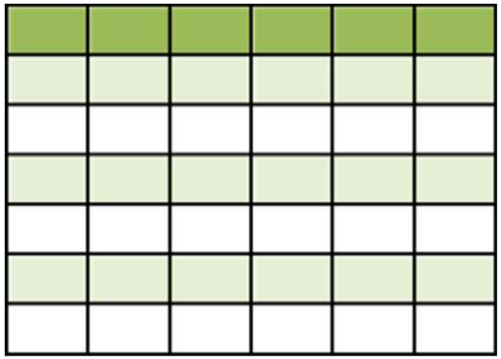
Virtual private database

```
create or replace function vpdfunc
(schemaname_in in varchar2
,tablename_in in varchar2) return varchar2
is
  l_returnvalue varchar2(2000);
begin
  case user
    when 'BU' then l_returnvalue := 'DEPTNO in (10,30)';
    when 'MP' then l_returnvalue := 'DEPTNO in (20,40)';
    else l_returnvalue := '1=1';
  end case;
  return l_returnvalue;
end;
```



Virtual private database

```
begin
  sys.dbms_rls.add_policy( object_schema   => 'DEMO'
                         , object_name     => 'EMP_VPD'
                         , policy_name     => 'VPDDEMO'
                         , function_schema => 'DEMO'
                         , policy_function => 'VPDFUNC' );
end;
```

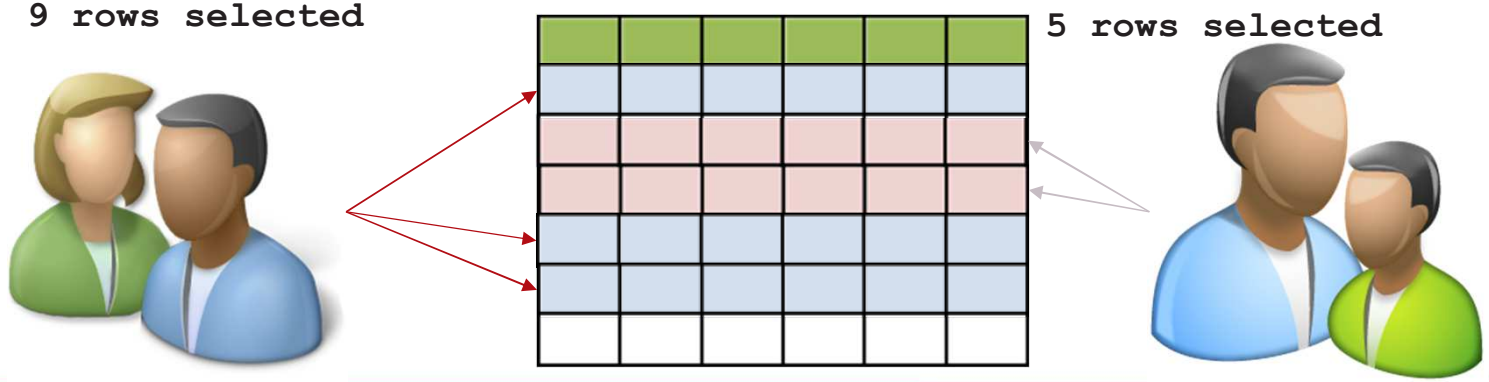


Virtual private database

```
begin
  sys.dbms_rls.add_policy( object_schema   => 'DEMO'
                         , object_name     => 'EMP_VPD'
                         , policy_name     => 'VPDDEMO'
                         , function_schema => 'DEMO'
                         , policy_function => 'VPDFUNC' );
end;
```

```
select empno, ename, deptno from demo.emp_vpd

EMPNO ENAME      DEPTNO
----- ---------- ------
 7499 ALLEN          30
…
 7934 MILLER         10
9 rows selected
```

```
EMPNO ENAME      DEPTNO
----- ---------- ------
 7369 SMITH          20
…
 7902 FORD           20
5 rows selected
```



VirtualPrivateDatabase.sql

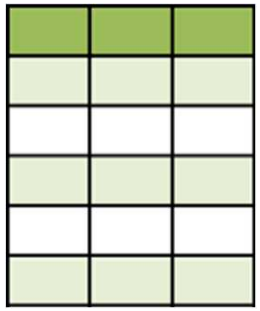

Real Application Security
Key Vault
Label Security
Database Vault
Advanced Security
a Masking
Subsetting
Unified and Conditional Auditing
Traditional Database Auditing
Audit Vault and Database Firewall

Performance increase

Query result cache

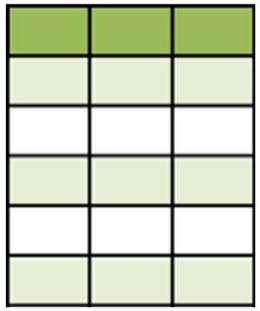
```
create table QueryResultCacheTable
( id number
)
```

Query result cache



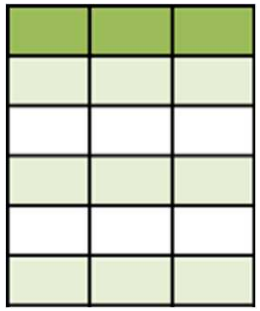
```
insert into QueryResultCacheTable(id) values (1);
insert into QueryResultCacheTable(id) values (2);
insert into QueryResultCacheTable(id) values (3);
insert into QueryResultCacheTable(id) values (4);
insert into QueryResultCacheTable(id) values (5);

create or replace function veryslowfunction(id_in in
queryresultcachetable.id%type)
  return queryresultcachetable.id%type as
begin
  dbms_lock.sleep(1);
  return id_in;
end;

set timing on
```

Query result cache



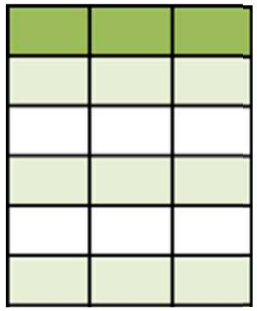
```
select VerySlowFunction(qrct.id)     Executed in 5.88 seconds
  from QueryResultCacheTable qrct

select /*+ result_cache */           Executed in 11.664 seconds
 veryslowfunction(qrct.id)
  from queryresultcachetable qrct

select /*+ result_cache */           Executed in 0.035 seconds
 veryslowfunction(qrct.id)
  from queryresultcachetable qrct
```

Query result cache



```
create or replace function veryslowfunction(id_in in
queryresultcachetable.id%type)
  return queryresultcachetable.id%type deterministic as
begin
  dbms_lock.sleep(1);
  return id_in;
end;
```

Query result cache



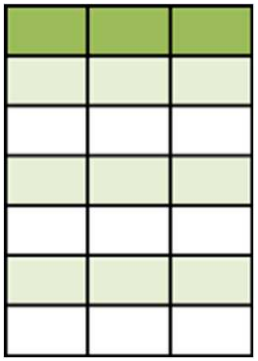

```
select /*+ result_cache */          Executed in 5.507 seconds
 veryslowfunction(qrct.id)
  from queryresultcachetable qrct

select /*+ result_cache */          Executed in 0.038 seconds
 veryslowfunction(qrct.id)
  from queryresultcachetable qrct
```

QueryResultCache.sql

Query result cache

```
create table FunctionResultCacheTable
( id number
)
```

Query result cache



```
insert into FunctionResultCacheTable(id) values (1);
insert into FunctionResultCacheTable(id) values (2);
insert into FunctionResultCacheTable(id) values (3);
insert into FunctionResultCacheTable(id) values (1);
insert into FunctionResultCacheTable(id) values (2);
insert into FunctionResultCacheTable(id) values (3);
insert into FunctionResultCacheTable(id) values (1);
insert into FunctionResultCacheTable(id) values (2);
insert into FunctionResultCacheTable(id) values (3);
insert into FunctionResultCacheTable(id) values (1);
```

Query result cache

```
create or replace function veryslowfunction(id_in in
FunctionResultCacheTable.id%type)
  return FunctionResultCacheTable.id%type                as
begin
  dbms_lock.sleep(1);
  return id_in;
end;
```

```
select veryslowfunction(frct.id)
  from FunctionResultCacheTable frct
```

```
Elapsed: 00:00:10.28                Elapsed: 00:00:10.18
```

Query result cache

```
create or replace function veryslowfunction(id_in in
FunctionResultCacheTable.id%type)
  return FunctionResultCacheTable.id%type deterministic as
begin
  dbms_lock.sleep(1);
  return id_in;
end;
```

```
select veryslowfunction(frct.id)
  from FunctionResultCacheTable frct
```

```
Elapsed: 00:00:10.28
Elapsed: 00:00:04.09
```

```
Elapsed: 00:00:10.18
Elapsed: 00:00:04.16
```

Query result cache

```
create or replace function veryslowfunction(id_in in
FunctionResultCacheTable.id%type)
  return FunctionResultCacheTable.id%type result_cache  as
begin
  dbms_lock.sleep(1);
  return id_in;
end;
```

```
select veryslowfunction(frct.id)
  from FunctionResultCacheTable frct
```

```
Elapsed: 00:00:10.28
Elapsed: 00:00:04.09
Elapsed: 00:00:03.15
Elapsed: 00:00:00.00
```

```
Elapsed: 00:00:10.18
Elapsed: 00:00:04.16
Elapsed: 00:00:00.00
```

FunctionResultCache.sql

Special SQL features

Special SQL features

Retrieve the top-3 earning employees

```
select e.empno
      ,e.ename
      ,e.sal
  from demo.emp e
 where rownum < 4
 order by e.sal desc
```

```
EMPNO ENAME             SAL
----- ---------- ---------
 7499 ALLEN         1600.00
 7521 WARD          1250.00
 7369 SMITH          800.00
```

Special SQL features

Retrieve the top-3 earning employees

```
select e.empno
      ,e.ename
      ,e.sal
  from (select e2.empno
              ,e2.ename
              ,e2.sal
          from demo.emp e2
         order by e2.sal desc) e
 where rownum < 4
```

```
EMPNO ENAME             SAL
----- ---------- ---------
 7839 KING          5000.00
 7788 SCOTT         3000.00
 7902 FORD          3000.00
```

Special SQL features

Retrieve the top-3 earning employees

```
with orderedbysal as
 (select e.empno
        ,e.ename
        ,e.sal

    from demo.emp e
   order by e.sal desc)
select obs.empno
      ,obs.ename
      ,obs.sal
  from orderedbysal obs
 where rownum < 4
```

Syntax Candy this is rewritten as

```
select e.empno
      ,e.ename
      ,e.sal
  from demo.emp e
 order by e.sal desc
 fetch first 3 rows only
```

```
EMPNO ENAME             SAL
----- ---------- ---------
 7839 KING          5000.00
 7788 SCOTT         3000.00
 7902 FORD          3000.00

EMPNO ENAME             SAL
----- ---------- ---------
 7839 KING          5000.00
 7788 SCOTT         3000.00
 7902 FORD          3000.00
```

Special SQL features

Retrieve the next 3 topearning employees

```
with orderedbysal as
 (select e.empno
        ,e.ename
        ,e.sal
        ,rownum rn
    from demo.emp e
   order by e.sal desc)
select obs.empno
      ,obs.ename
      ,obs.sal
  from orderedbysal obs
 where obs.rn between 4 and 6
 order by obs.rn
```

```
EMPNO ENAME             SAL
----- ---------- ---------
 7566 JONES         2975.00
 7654 MARTIN        1250.00
 7698 BLAKE         2850.00
```

Special SQL features

Retrieve the next 3 topearning employees

```
with orderedbysal as
 (select e.empno
        ,e.ename
        ,e.sal
    from demo.emp e
   order by e.sal desc)
, orderedwithrownum as
 (select obs.empno
       , obs.ename
       , obs.sal
       , rownum rn
    from orderedbysal obs)
select obr.empno
      ,obr.ename
      ,obr.sal
  from orderedwithrownum obr
 where obr.rn between 4 and 6
 order by obr.rn
```

```
select e.empno
      ,e.ename
      ,e.sal
  from demo.emp e
 order by e.sal desc
offset 3 rows
 fetch next 3 rows only
```

```
EMPNO ENAME             SAL
----- ---------- ---------
 7566 JONES         2975.00
 7698 BLAKE         2850.00
 7782 CLARK         2450.00

EMPNO ENAME             SAL
----- ---------- ---------
 7566 JONES         2975.00
 7698 BLAKE         2850.00
 7782 CLARK         2450.00
```

paginating.sql

Special SQL features

Build PL/SQL in SQL

```
create or replace function wavey(string_in in varchar2)
  return varchar2
is
  l_returnvalue varchar2(30);
begin
  for indx in 1 .. length(string_in) loop
    l_returnvalue := l_returnvalue ||
      case mod(indx, 2)
        when 0 then upper(substr(string_in, indx, 1))
               else lower(substr(string_in, indx, 1))
      end;
  end loop;
  return l_returnvalue;
end;


Function created
```

Special SQL features

Build PL/SQL in SQL

```
select e.empno, e.ename, wavey(e.ename) wavey_ename, e.sal
  from demo.emp e
```

EMPNO	ENAME	WAVEY_ENAME	SAL
7369	SMITH	sMiTh	800.00
7499	ALLEN	aLlEn	1600.00
7521	WARD	wArD	1250.00
7566	JONES	jOnEs	2975.00
7654	MARTIN	mArTiN	1250.00
7698	BLAKE	bLaKe	2850.00
7782	CLARK	cLaRk	2450.00
7788	SCOTT	sCoTt	3000.00
7839	KING	kInG	5000.00
7844	TURNER	tUrNeR	1500.00
7876	ADAMS	aDaMs	1100.00
7900	JAMES	jAmEs	950.00
7902	FORD	fOrD	3000.00
7934	MILLER	mIlLeR	1300.00

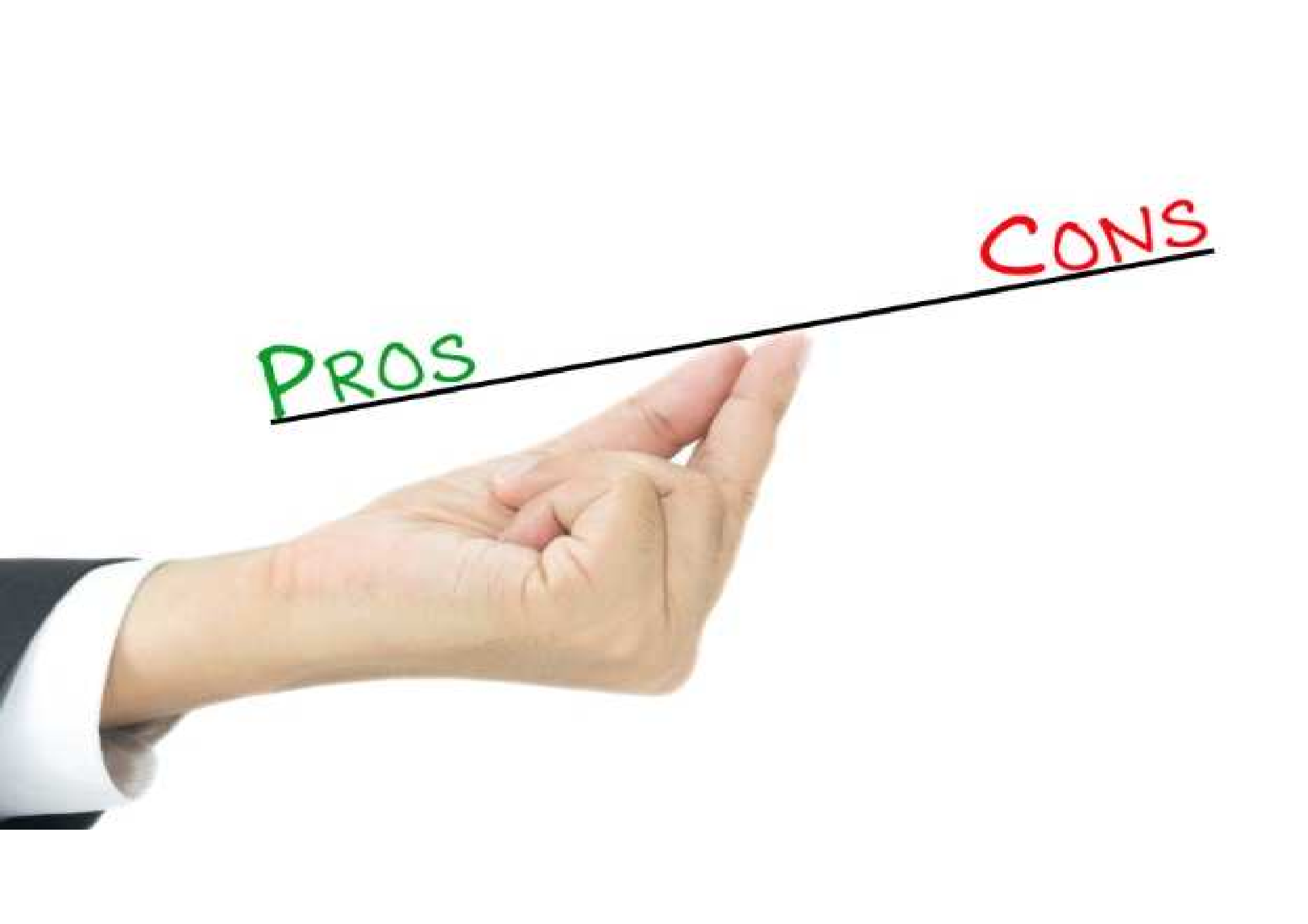
14 rows selected

Special SQL features

```
with function wavey(string_in in varchar2) return varchar2
  is
    l_returnvalue varchar2(30);
  begin
    for indx in 1 .. length(string_in) loop
      l_returnvalue := l_returnvalue ||
          case mod(indx, 2)
            when 1 then upper(substr(string_in, indx, 1))
                   else lower(substr(string_in, indx, 1))
          end;
    end loop;
    return l_returnvalue;
  end;
 select e.empno, e.ename, wavey(e.ename) wavey_ename, e.sal
   from demo.emp e
```

```
EMPNO ENAME      WAVEY_ENAME         SAL
----- ---------- ------------ ---------
 7369 SMITH      SmItH             800.00
 7499 ALLEN      AlLeN            1600.00
...
 7934 MILLER     MiLlEr           1300.00
14 rows selected
```

plsql_in_with.sql

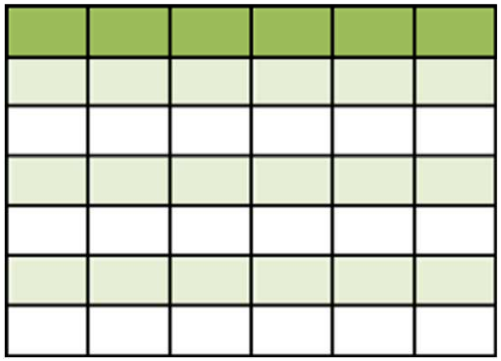

PROS
CONS

Virtual Columns

Virtual Columns

```
create table demo.emp_income
(
  empno     number(4) not null
, ename     varchar2(10)
, sal       number(7,2)
, comm      number(7,2)
, deptno    number(2)
, income    number(7,2)
                    generated always as
                   (demo.emp_income.sal +
                    coalesce(demo.emp_income.comm,0)) virtual
)
```

Virtual Columns



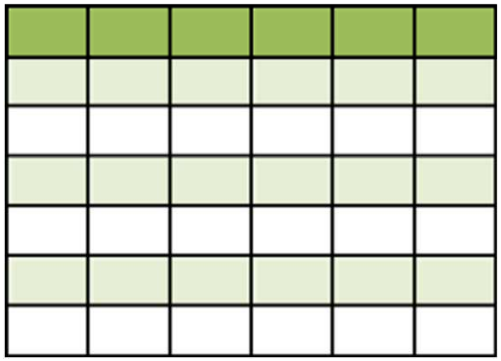
```
insert into demo.emp_income(empno, ename, sal, deptno, comm)
                      values (7499, 'ALLEN',  1600, 30, 300);
insert into demo.emp_income(empno, ename, sal, deptno, comm)
                      values (7521, 'WARD',   1250, 30, 500);
insert into demo.emp_income(empno, ename, sal, deptno, comm)
                     values (7654, 'MARTIN', 1250, 30, 1400);
insert into demo.emp_income(empno, ename, sal, deptno, comm)
                     values (7698, 'BLAKE',  2850, 30, null);
insert into demo.emp_income(empno, ename, sal, deptno, comm)
                        values (7844, 'TURNER', 1500, 30, 0);
insert into demo.emp_income(empno, ename, sal, deptno, comm)
                     values (7900, 'JAMES',  950,  30, null);
```

Virtual Columns

```
DEMO@demo> insert into demo.emp_income
  2    (empno, ename, sal, deptno, comm, income)
  3  values
  4    (2912, 'BAREL', 3000, 30, 2000, 5000)
  5  /
(empno, ename, sal, deptno, comm, income)
                                    *
ERROR at line 2:
ORA-54013: INSERT operation disallowed on virtual columns
```

```
insert into demo.emp_in
                        values (
insert into demo.emp_income(empno, ename,
                        values (7521, 'WARD',
insert into demo.emp_income(empno, ename, sal, deptno,
                       values (7654, 'MARTIN', 1250, 30, 1400);
insert into demo.emp_income(empno, ename, sal, deptno, comm)
                       values (7698, 'BLAKE',  2850, 30, null);
insert into demo.emp_income(empno, ename, sal, deptno, comm)
                          values (7844, 'TURNER', 1500, 30, 0);
insert into demo.emp_income(empno, ename, sal, deptno, comm)
                       values (7900, 'JAMES',  950,  30, null);
```

Virtual Columns



```
select *
  from demo.emp_income e
```

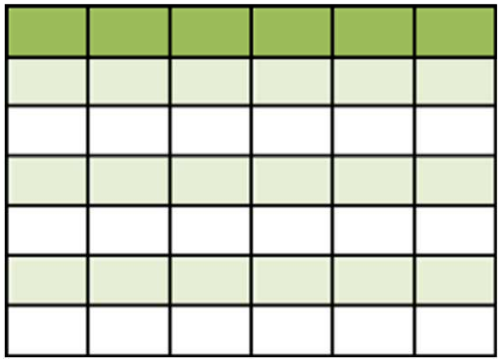
```
 EMPNO ENAME             SAL       COMM   DEPTNO     INCOME
------ ---------- --------- --------- -------- ---------
  7499 ALLEN        1600.00    300.00       30    1900.00
  7521 WARD         1250.00    500.00       30    1750.00
  7654 MARTIN       1250.00   1400.00       30    2650.00
  7698 BLAKE        2850.00                 30    2850.00
  7844 TURNER       1500.00        .00       30    1500.00
  7900 JAMES         950.00                 30     950.00

6 rows selected.
```

VirtualColumnIncome.sql

Virtual Columns

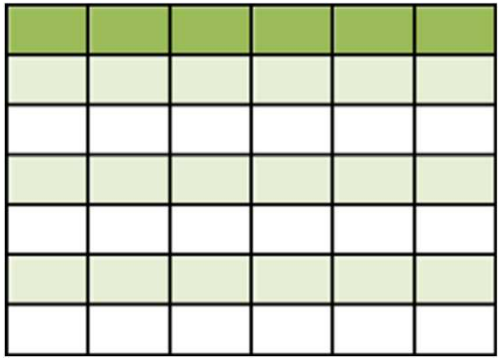
```
create table demo.emp_nuke
(
  empno  number(4) not null
, ename  varchar2(10)
, sal    number(7,2)
, deptno number(2)
, blowitup number generated always as (1/0) virtual
)
```

Virtual Columns



```
insert into demo.emp_nuke(empno, ename, sal, deptno)
                    values (7499, 'ALLEN',  1600, 30);
insert into demo.emp_nuke(empno, ename, sal, deptno)
                    values (7521, 'WARD',   1250, 30);
insert into demo.emp_nuke(empno, ename, sal, deptno)
                    values (7654, 'MARTIN', 1250, 30);
insert into demo.emp_nuke(empno, ename, sal, deptno)
                    values (7698, 'BLAKE',  2850, 30);
insert into demo.emp_nuke(empno, ename, sal, deptno)
                    values (7844, 'TURNER', 1500, 30);
insert into demo.emp_nuke(empno, ename, sal, deptno)
                    values (7900, 'JAMES',  950,  30);
```

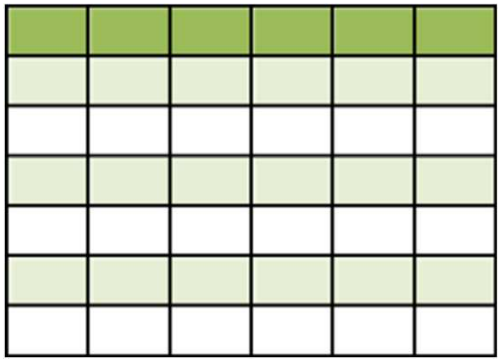
Virtual Columns



```
select *
  from demo.emp_nuke e
```

```
select *
 *
ERROR at line 1:
ORA-01476: divisor is equal to zero
```

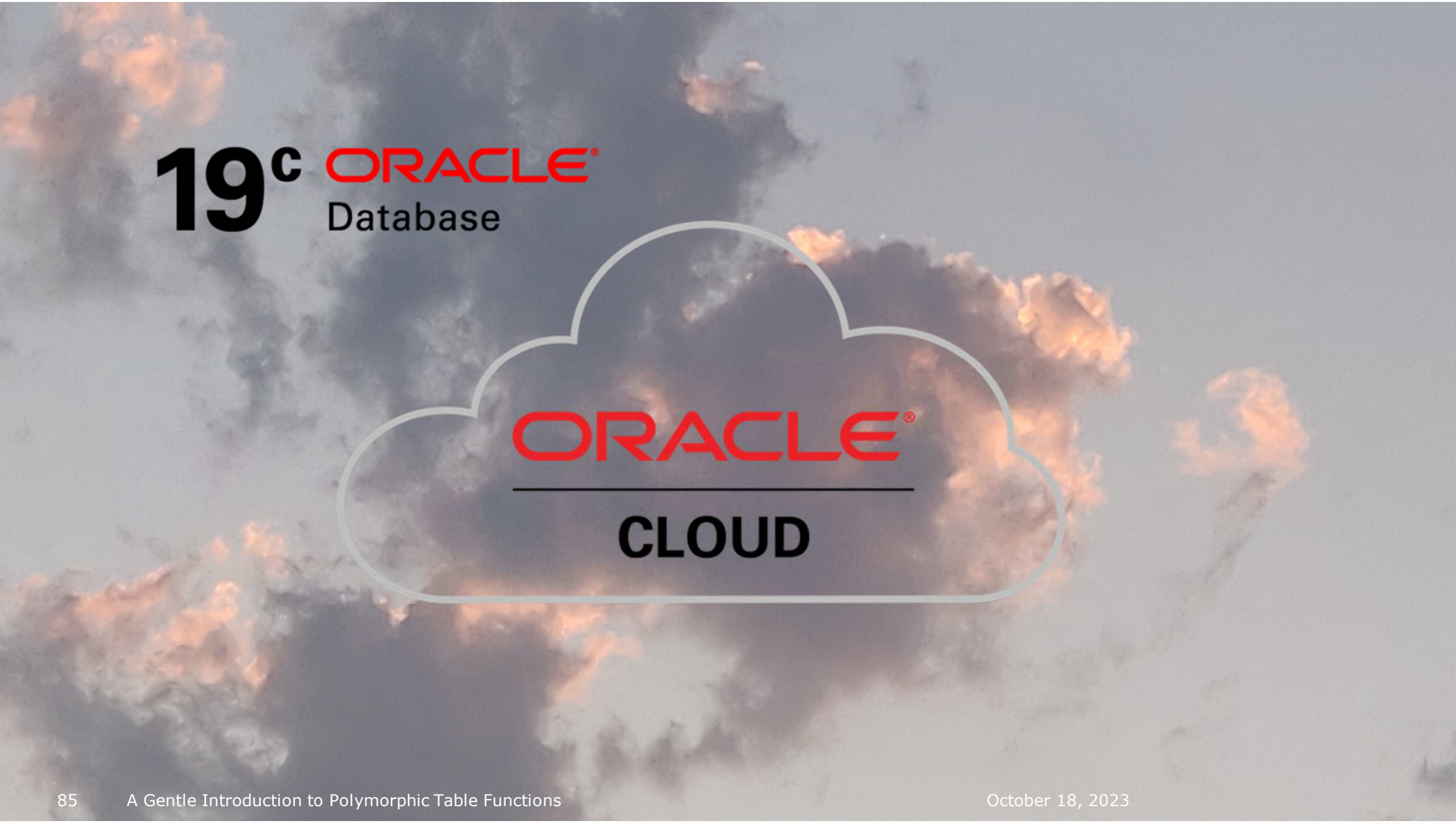
Virtual Columns

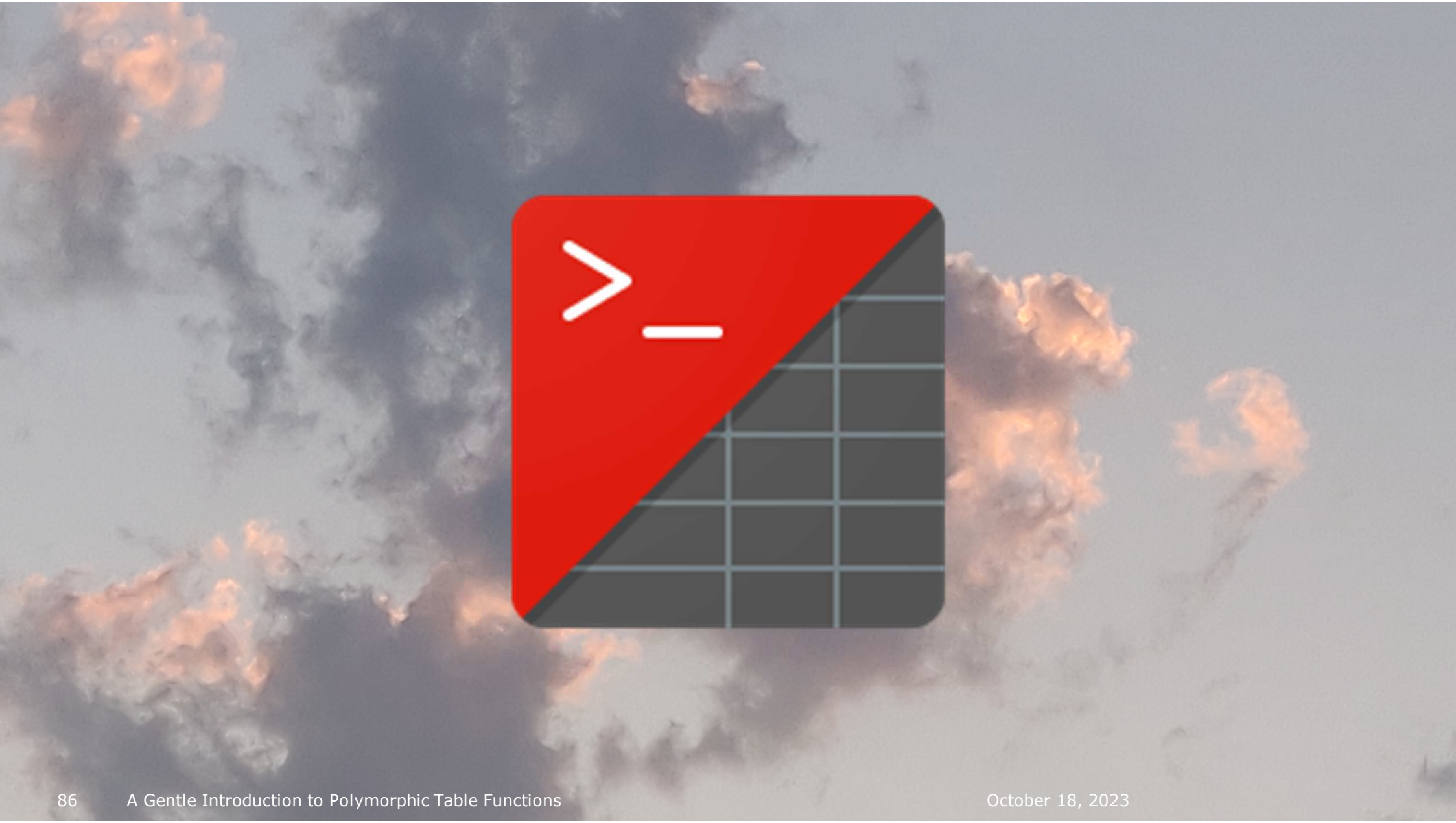


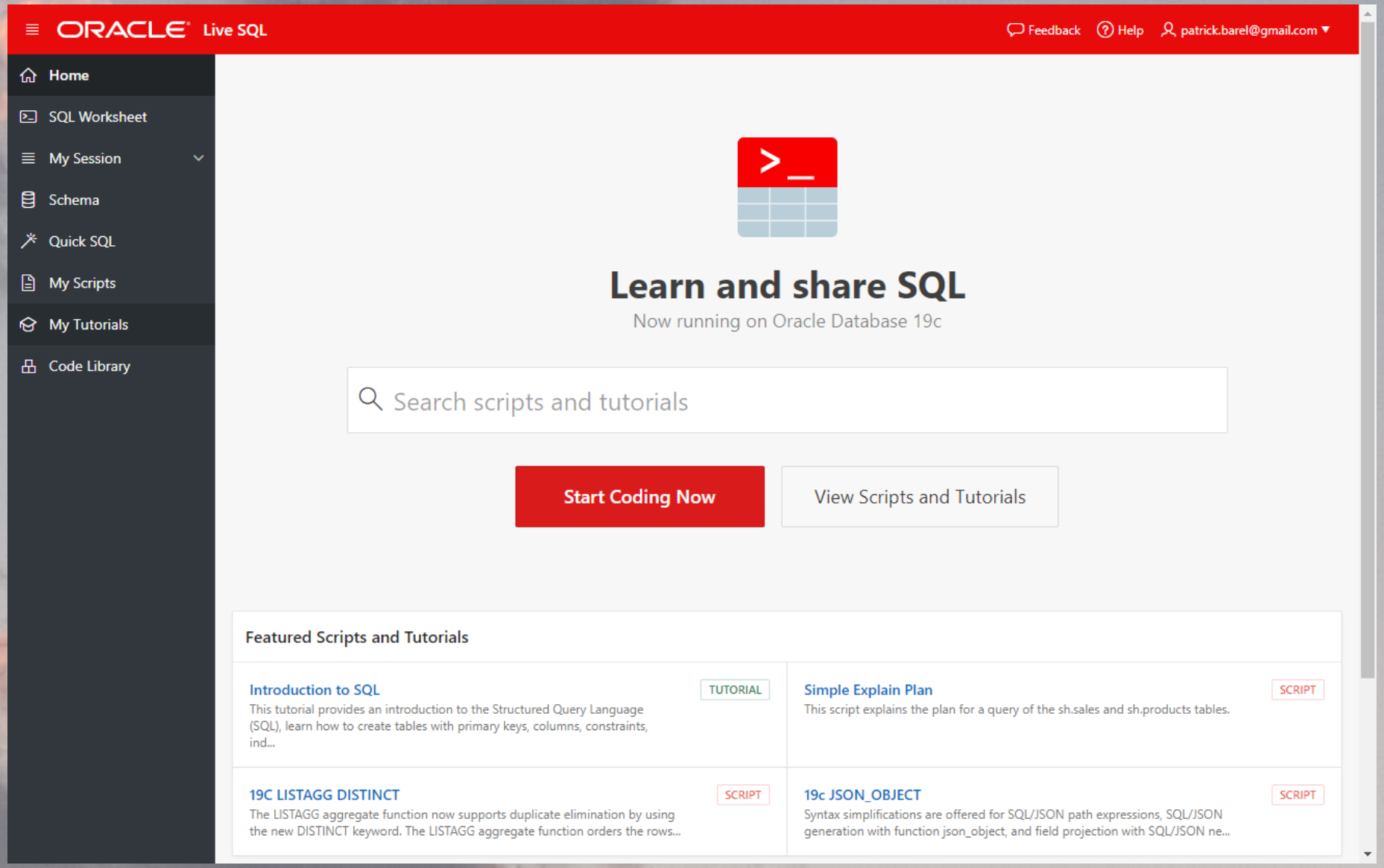
```
select e.empno, e.ename, e.sal, e.deptno
  from demo.emp_nuke e
```

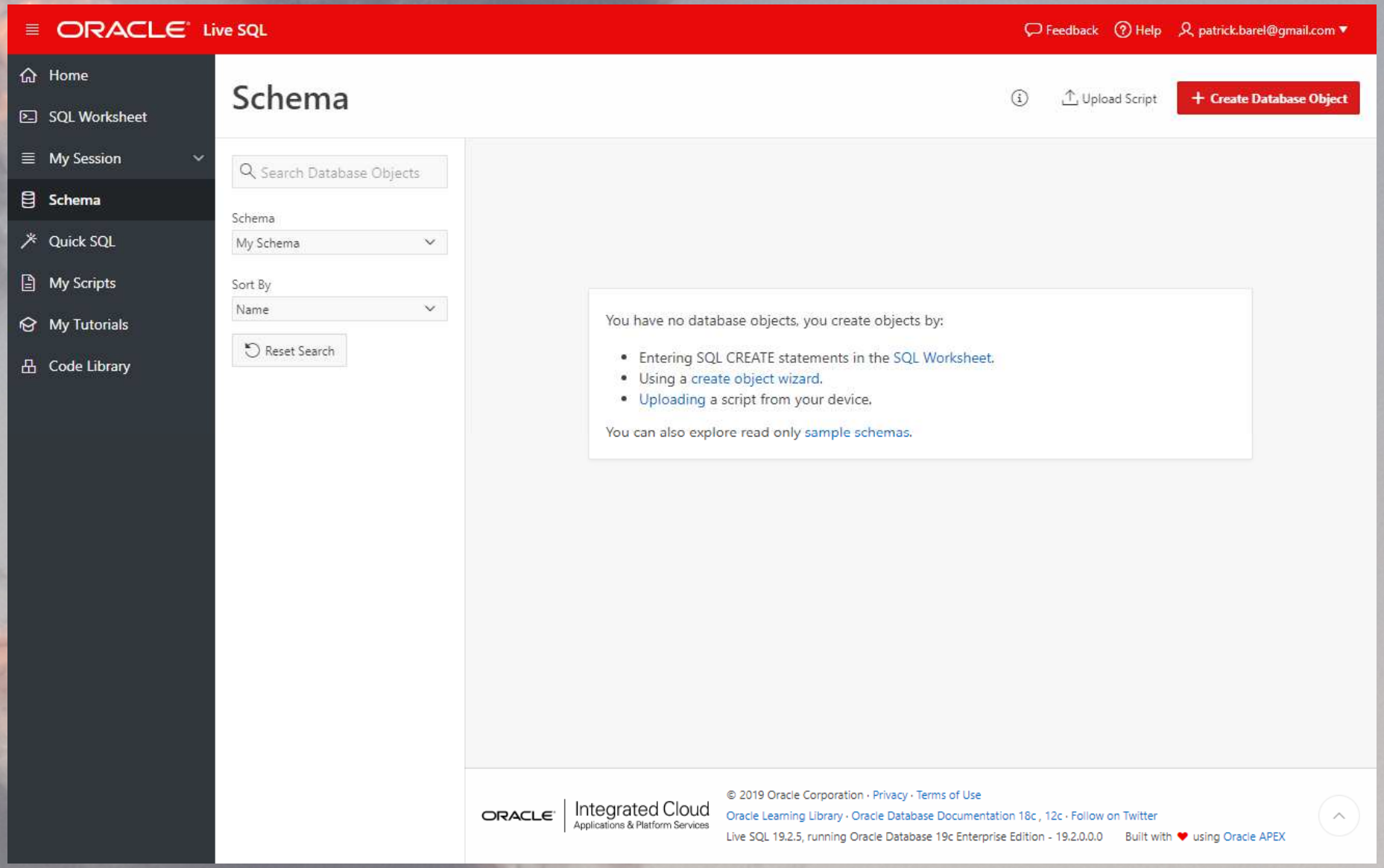
```
 EMPNO ENAME                SAL  DEPTNO
------ ---------- --------- -------
  7499 ALLEN          1600.00      30
  7521 WARD           1250.00      30
  7654 MARTIN         1250.00      30
  7698 BLAKE          2850.00      30
  7844 TURNER         1500.00      30
  7900 JAMES           950.00      30

6 rows selected.
```

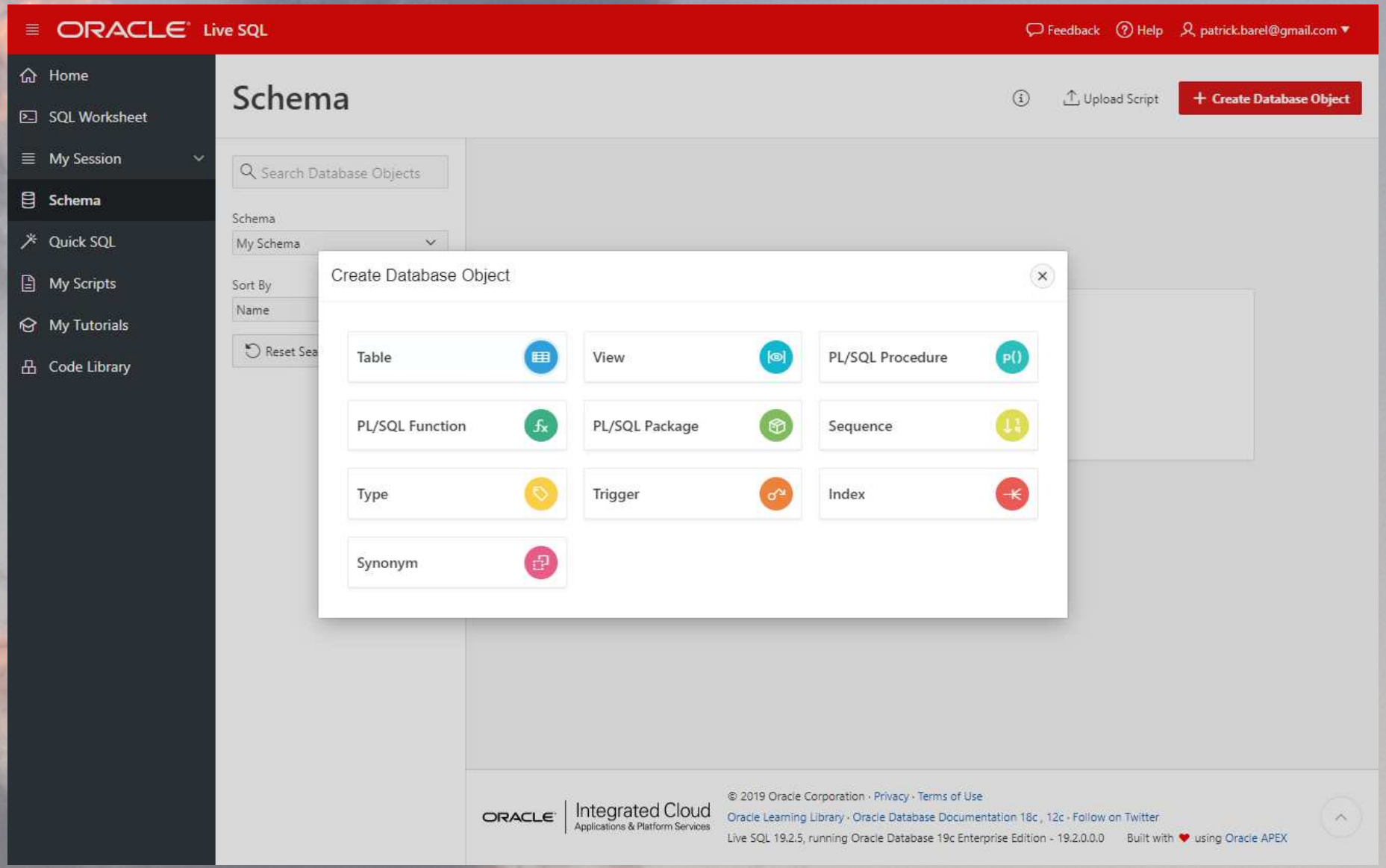
VirtualColumnPreventSelectStar.sql

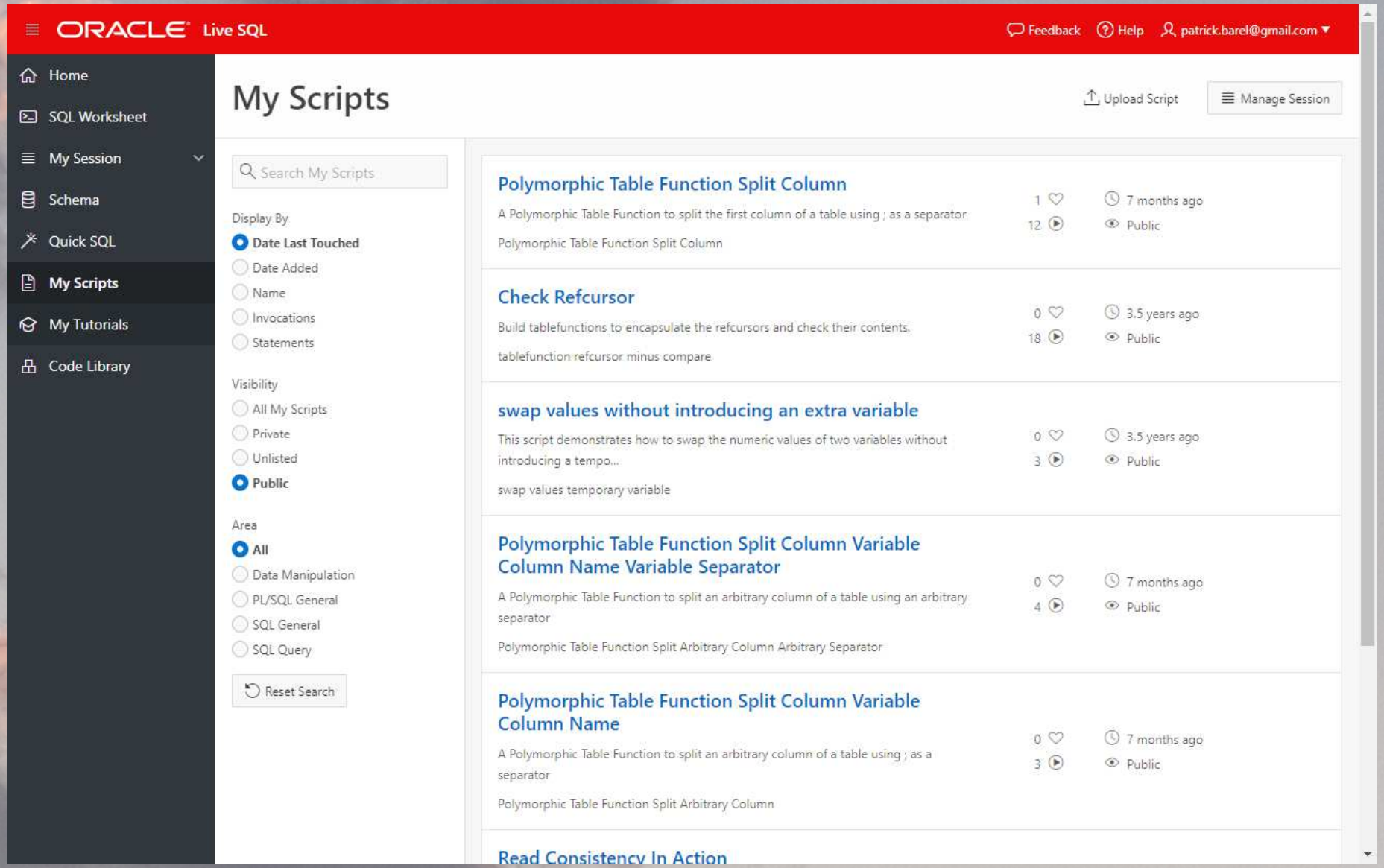

19c
ORACLE
Database
ORACLE
CLOUD

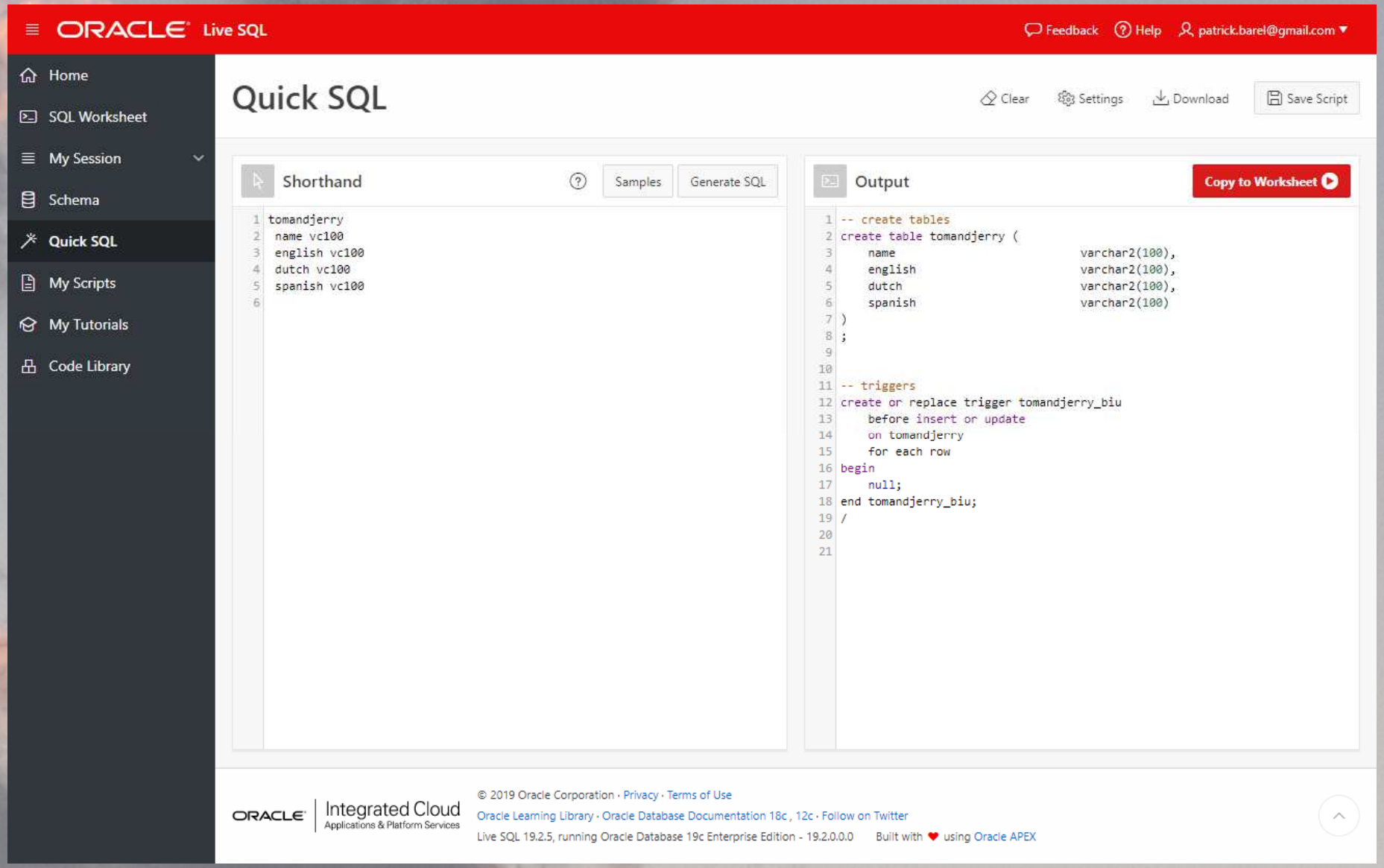



ORACLE Live SQL
Feedback
Help
patrick.barel@gmail.com
Home
SQL Worksheet
My Session
Schema
Quick SQL
My Scripts
My Tutorials
Code Library
Learn and share SQL
Now running on Oracle Database 19c
Search scripts and tutorials
Start Coding Now
View Scripts and Tutorials
Featured Scripts and Tutorials
Introduction to SQL
TUTORIAL
This tutorial provides an introduction to the Structured Query Language (SQL), learn how to create tables with primary keys, columns, constraints, ind...
Simple Explain Plan
SCRIPT
This script explains the plan for a query of the sh.sales and sh.products tables.
19C LISTAGG DISTINCT
SCRIPT
The LISTAGG aggregate function now supports duplicate elimination by using the new DISTINCT keyword. The LISTAGG aggregate function orders the rows...
19c JSON_OBJECT
SCRIPT
Syntax simplifications are offered for SQL/JSON path expressions, SQL/JSON generation with function json_object, and field projection with SQL/JSON ne...


ORACLE Live SQL
Feedback
Help
patrick.barel@gmail.com
Home
SQL Worksheet
My Session
Schema
Quick SQL
My Scripts
My Tutorials
Code Library
Schema
Upload Script
+ Create Database Object
Search Database Objects
Schema
My Schema
Sort By
Name
Reset Search
You have no database objects, you create objects by:
Entering SQL CREATE statements in the SQL Worksheet.
Using a create object wizard.
Uploading a script from your device.
You can also explore read only sample schemas.
ORACLE Integrated Cloud
Applications & Platform Services
© 2019 Oracle Corporation · Privacy · Terms of Use
Oracle Learning Library · Oracle Database Documentation 18c , 12c · Follow on Twitter
Live SQL 19.2.5, running Oracle Database 19c Enterprise Edition - 19.2.0.0.0
Built with ❤ using Oracle APEX


ORACLE Live SQL
Feedback
Help
patrick.barel@gmail.com
Home
SQL Worksheet
My Session
Schema
Quick SQL
My Scripts
My Tutorials
Code Library
Schema
Upload Script
+ Create Database Object
Search Database Objects
Schema
My Schema
Sort By
Name
Reset Sea
Create Database Object
Table
View
PL/SQL Procedure
PL/SQL Function
PL/SQL Package
Sequence
Type
Trigger
Index
Synonym
ORACLE Integrated Cloud
Applications & Platform Services
© 2019 Oracle Corporation · Privacy · Terms of Use
Oracle Learning Library · Oracle Database Documentation 18c , 12c · Follow on Twitter
Live SQL 19.2.5, running Oracle Database 19c Enterprise Edition - 19.2.0.0.0
Built with ❤ using Oracle APEX


ORACLE Live SQL
Feedback
Help
patrick.barel@gmail.com
Home
SQL Worksheet
My Session
Schema
Quick SQL
My Scripts
My Tutorials
Code Library
My Scripts
Upload Script
Manage Session
Search My Scripts
Display By
Date Last Touched
Date Added
Name
Invocations
Statements
Visibility
All My Scripts
Private
Unlisted
Public
Area
All
Data Manipulation
PL/SQL General
SQL General
SQL Query
Reset Search
Polymorphic Table Function Split Column
A Polymorphic Table Function to split the first column of a table using ; as a separator
Polymorphic Table Function Split Column
1
12
7 months ago
Public
Check Refcursor
Build tablefunctions to encapsulate the refcursors and check their contents.
tablefunction refcursor minus compare
0
18
3.5 years ago
Public
swap values without introducing an extra variable
This script demonstrates how to swap the numeric values of two variables without introducing a tempo...
swap values temporary variable
0
3
3.5 years ago
Public
Polymorphic Table Function Split Column Variable Column Name Variable Separator
A Polymorphic Table Function to split an arbitrary column of a table using an arbitrary separator
Polymorphic Table Function Split Arbitrary Column Arbitrary Separator
0
4
7 months ago
Public
Polymorphic Table Function Split Column Variable Column Name
A Polymorphic Table Function to split an arbitrary column of a table using ; as a separator
Polymorphic Table Function Split Arbitrary Column
0
3
7 months ago
Public
Read Consistency In Action


ORACLE Live SQL
Feedback
Help
patrick.barel@gmail.com
Home
SQL Worksheet
My Session
Schema
Quick SQL
My Scripts
My Tutorials
Code Library
Quick SQL
Clear
Settings
Download
Save Script
Shorthand
Samples
Generate SQL
tomandjerry
name vc100
english vc100
dutch vc100
spanish vc100
Output
Copy to Worksheet
-- create tables
create table tomandjerry (
name varchar2(100),
english varchar2(100),
dutch varchar2(100),
spanish varchar2(100)
)
;
-- triggers
create or replace trigger tomandjerry_biu
before insert or update
on tomandjerry
for each row
begin
null;
end tomandjerry_biu;
/
ORACLE Integrated Cloud
Applications & Platform Services
© 2019 Oracle Corporation · Privacy · Terms of Use
Oracle Learning Library · Oracle Database Documentation 18c , 12c · Follow on Twitter
Live SQL 19.2.5, running Oracle Database 19c Enterprise Edition - 19.2.0.0.0
Built with using Oracle APEX

Oracle Cloud Infrastructure

New Free Tier

oracle.com/cloud/free

Always Free

Services you can use for unlimited time

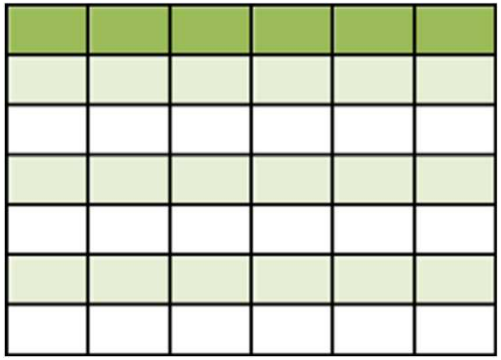
+

30-Day Free Trial

Free credits you can use for more services

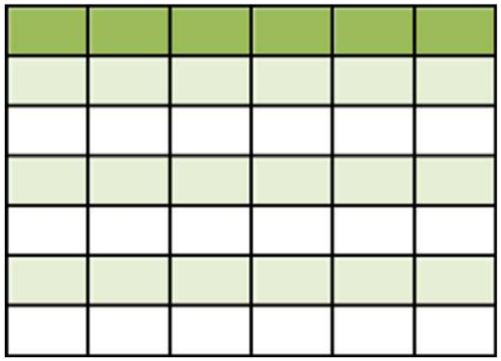
Identity columns (12c)

Identity columns (12c)



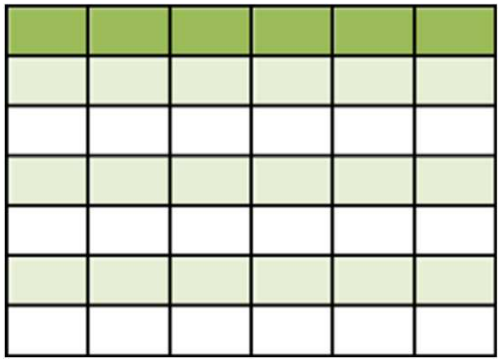
```
create table tablewithuniquecolumn
( theuniquecolumn number(10) not null
, foo varchar2(30)
, bar varchar2(30)
)
```

Identity columns (12c)



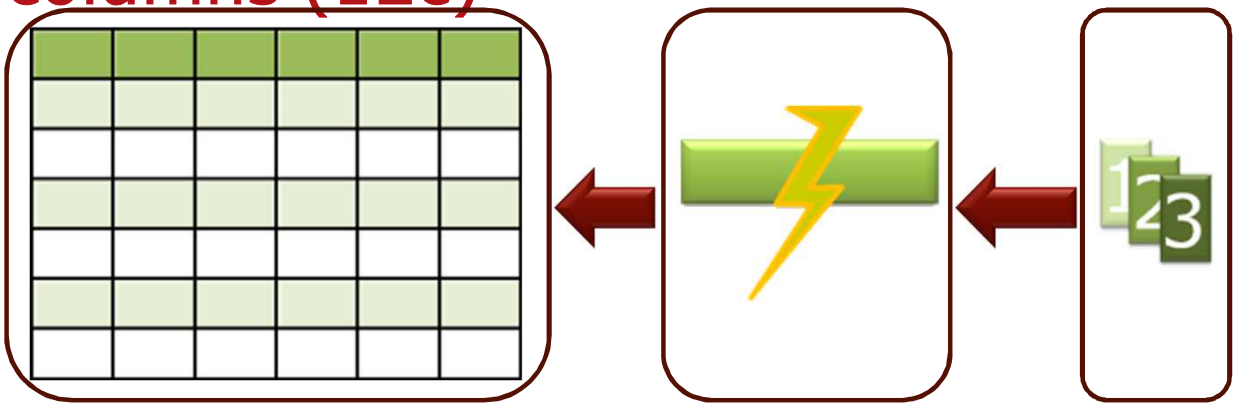
```
create sequence sequenceforuniquecolumn start with 1 nocache
```

Identity columns (12c)



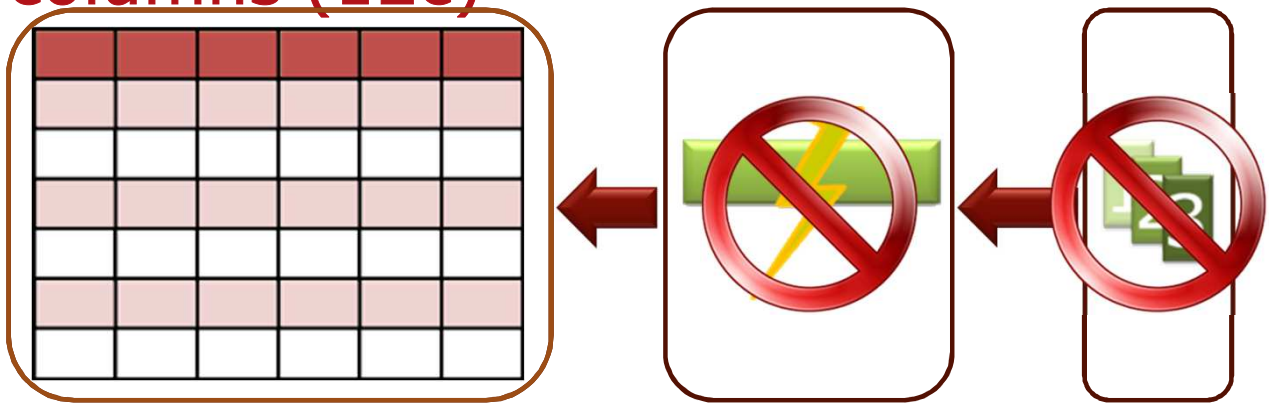
```
create or replace trigger makesureitsfilled
  before insert or update on tablewithuniquecolumn
  for each row
begin
  if :new.theuniquecolumn is null then
    select sequenceforuniquecolumn.nextval
      into :new.theuniquecolumn
      from dual;
  end if;
end;
```

Identity columns (12c)



```
create or replace trigger makesureitsfilled
  before insert or update on tablewithuniquecolumn
  for each row
begin
  if :new.theuniquecolumn is null then

    :new.theuniquecolumn := sequenceforuniquecolumn.nextval;

  end if;
end;
```

Identity columns (12c)



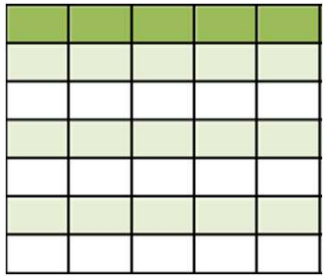
```
create table tablewithidentitycolumn
( theidentitycolumn number(10) generated always as identity
, foo varchar2(30)
, bar varchar2(30)
)
```

identity_columns.sql

Identity columns (12c)

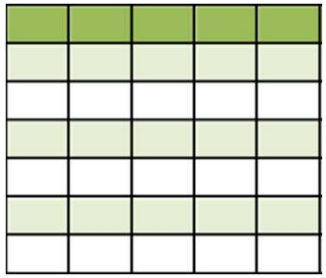
Invisible columns (12c)

Invisible columns (12c)



```
create table Show
( I varchar2(30)
, am varchar2(30)
, the varchar2(30)
, important varchar2(30)
, man varchar2(30)
)
```

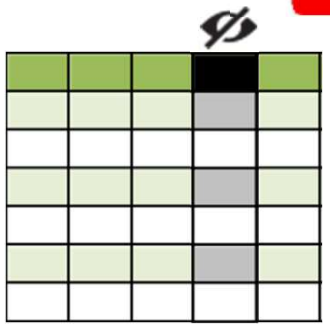
Invisible columns (12c)



```
I AM THE IMPORTANT MAN
- -- --- --------- ---
I am the important man
```

```
select * from show
```

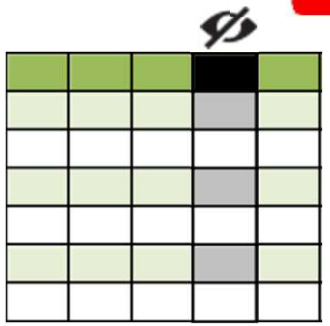
Invisible columns (12c)



```
I AM THE IMPORTANT MAN
- -- --- --------- ---
I am the important man
```

```
select * from show

alter table show modify important invisible
```

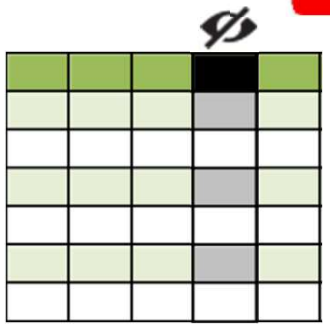
Invisible columns (12c)



```
I AM THE MAN IMPORTANT MAN
- -- --- --------- ---
I am the man important man
```

```
select * from show

alter table show modify important invisible
```

Invisible columns (12c)

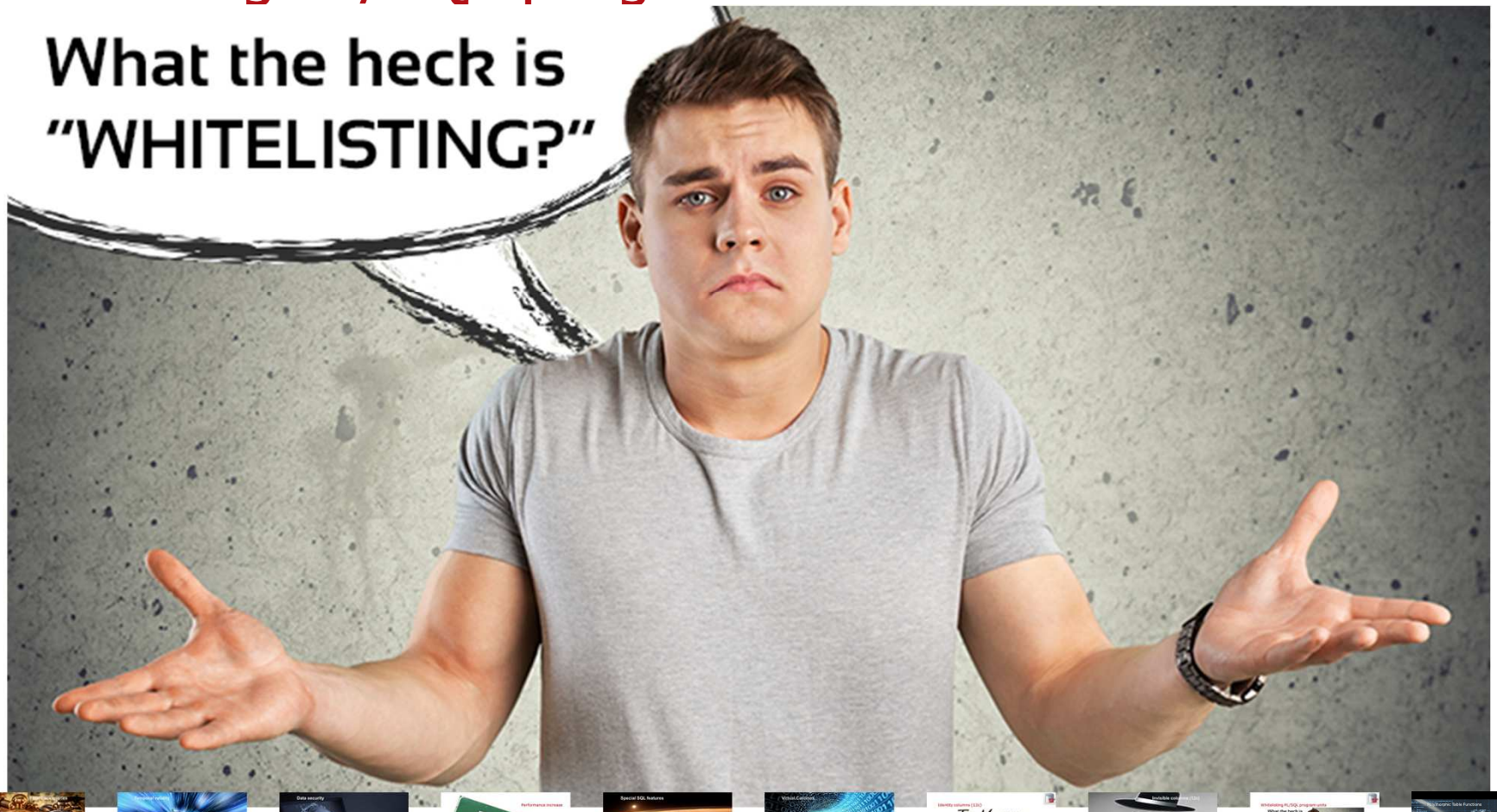


```
I AM THE MAN IMPORTANT MAN
- -- --- --------- ---
I am the man important man
```

```
select * from show

alter table show modify important invisible

select I, am, the, important, man
  from show
```

IAmTheInvisibleColumn.sql

Whitelisting PL/SQL program units



Whitelisting PL/SQL program units

Whitelisting PL/SQL program units


```
create or replace procedure onlywhitelisted
  accessible by (normalprocedure)
is
begin
  dbms_output.put_line('You are allowed');
end;
```

Whitelisting PL/SQL program units


```
create or replace procedure onlywhitelisted
  accessible by (normalprocedure)
is
begin
  dbms_output.put_line('You are allowed');
end;
```

```
create or replace procedure
  normalprocedure
is
begin
  dbms_output.put_line
   ('normalprocedure calls
     onlywhitelisted');
  onlywhitelisted;
end;
```

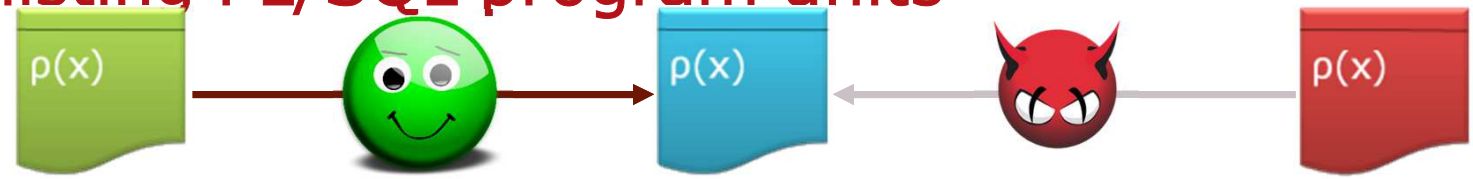
Whitelisting PL/SQL program units


```
create or replace procedure onlywhitelisted
  accessible by (normalprocedure)
is
begin
  dbms_output.put_line('You are allowed');
end;
```

```
create or replace procedure
  normalprocedure
is
begin
  dbms_output.put_line
   ('normalprocedure calls
     onlywhitelisted');
  onlywhitelisted;
end;
```

```
create or replace procedure
  evilprocedure
is
begin
  dbms_output.put_line
   ('evilprocedure calls
     onlywhitelisted');
  onlywhitelisted;
end;
```

Whitelisting PL/SQL program units



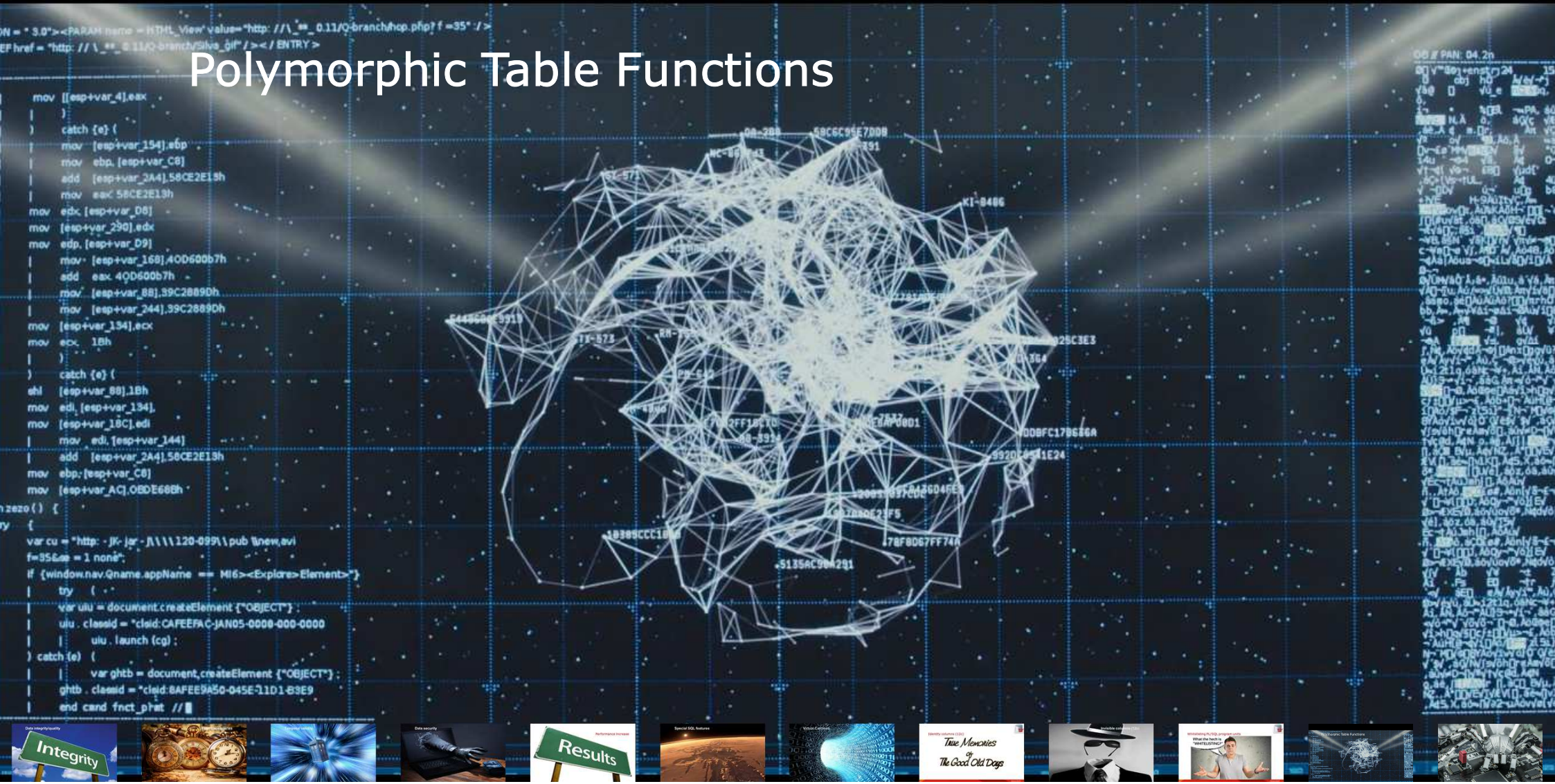

```
create or replace procedure onlywhitelisted
  accessible by (normalprocedure)
is
begin
  dbms_output.put_line('You are allowed');
end;
```

```
create or replace procedure
  normalprocedure
is
begin
  dbms_output.put_line
   ('normalprocedure calls
     onlywhitelisted');
  onlywhitelisted;
end;
```

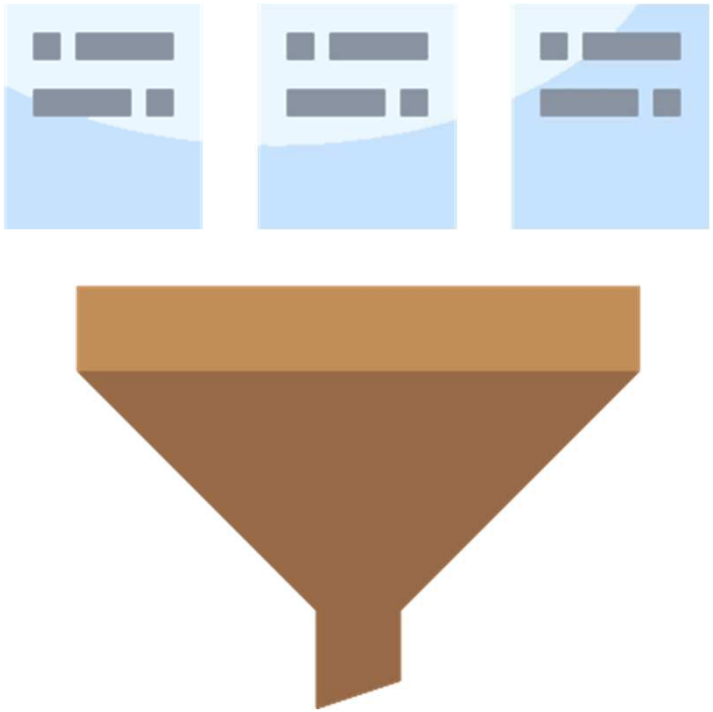
```
create or replace procedure
  evilprocedure
is
begin
  dbms_output.put_line
   ('evilprocedure calls
     onlywhitelisted');
  onlywhitelisted;
end;
```

whitelisting_12c.sql

Polymorphic Table Functions



Polymorphic Table Functions



Polymorphic Table Functions

```
select e.*,
       dbms_crypto.hash(src => utl_i18n.string_to_raw(to_char(EMPNO) || to_char(ENAME)
                                                      ,'AL32UTF8')
                        ,typ => 2) emphash
  from emp e
/
```

Polymorphic Table Functions

```
select d.*,
       dbms_crypto.hash(src => utl_i18n.string_to_raw(to_char(DEPTNO) || to_char(DNAME) ||
                                                      to_char(LOC)
                                                      ,'AL32UTF8')
                        ,typ => 2) depthash
  from dept d
/
```

Polymorphic Table Functions

```
select d.*,
       dbms_crypto.hash(src => utl_i18n.string_to_raw(to_char(DEPTNO) || to_char(DNAME) ||
                                                      to_char(LOC)
                                                      ,'AL32UTF8')
                        ,typ => 2) depthash
  from dept d
/
```

Polymorphic Table Functions

```
create or replace package hash_ptf is
  function describe(tab  in out dbms_tf.table_t
                   ,cols in dbms_tf.columns_t default null) return dbms_tf.describe_t;

  procedure fetch_rows;
end hash_ptf;
/
```

Polymorphic Table Functions

```
create or replace package body hash_ptf as
  subtype maxvarchar2 is varchar2(32767);
  function describe(tab  in out dbms_tf.table_t
                   ,cols in dbms_tf.columns_t default null) return dbms_tf.describe_t as
    -- metadata for column to add
    l_new_col dbms_tf.column_metadata_t;
    -- table of columns to add
    l_new_cols dbms_tf.columns_new_t; -- := DBMS_TF.COLUMNS_NEW_T();
  begin
    -- find the columns to redact
    -- mark them for_read, but not pass_through
    <<outer_loop>>
    for indx in tab.column.first .. tab.column.last loop
      if (cols is not null)
         and (cols.count > 0) then
        <<inner_loop>>
        for idx in 1 .. cols.count loop
          -- first mark every column not to be read, only to pass-through
          tab.column(indx).for_read := false;
          tab.column(indx).pass_through := true;
```

Polymorphic Table Functions

```
        l_total := l_total || dbms_tf.col_to_char(l_rowset(colindx), rowindx);
      else
        -- Catch all others
        dbms_output.put_line(q'[Columns of this type (]' ||
                             env.get_columns(colindx).type ||
                             q'[) are not supported (yet).]');
    end case;
  end loop; -- every column
  -- hash the total value
  l_hash(rowindx) := dbms_crypto.hash(src => utl_i18n.string_to_raw(l_total
                                                                    ,'AL32UTF8')
                                      ,typ => dbms_crypto.hash_md5);
  end loop; -- every row in the rowset
  dbms_tf.put_col(columnid => 1, collection => l_hash);
  -- add the newly populated columns to the rowset
  for indx in 1 .. l_putcolcount loop
    dbms_tf.put_col(columnid => indx, collection => l_newcolset(indx));
  end loop;
end;
end hash_ptf;
```

Polymorphic Table Functions

```
-- create a 'wrapper' function for the polymorphic table function
create or replace function hash_fnc(p_tbl      in table
                                   ,cols       columns default null) return table
  pipelined row polymorphic using hash_ptf;
/
```

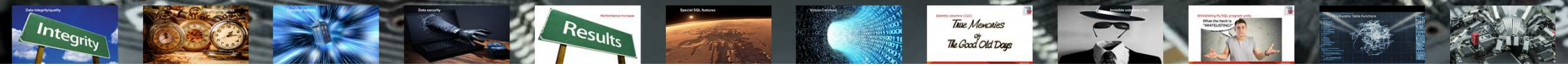
Polymorphic Table Functions

```
select empno, ename, job, mgr, hiredate, sal, comm, deptno, HashValue emphash
  from hash_fnc(emp, columns(empno, ename))
/
```

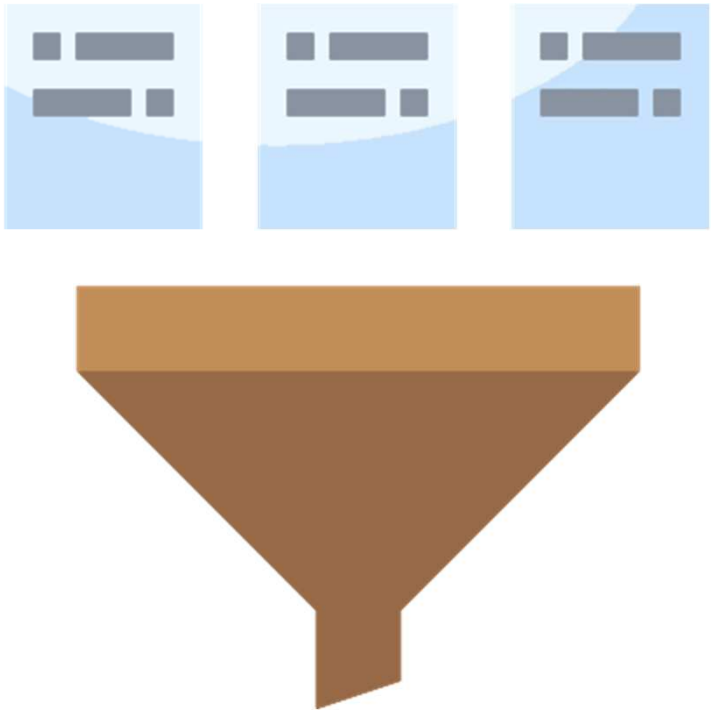
Polymorphic Table Functions

```
select deptno, dname, loc, HashValue depthash
  from hash_fnc(dept, columns(deptno, dname, loc))
/
DEPTNO DNAME          LOC            DEPTHASH
------ -------------- ------------- --------------------------------
    10 ACCOUNTING     NEW YORK      A7D6E3608EA91E1D71B58EFC14709298
    20 RESEARCH       DALLAS        93EF4EBE54185179277DA5B77792B3F5
    30 SALES          CHICAGO       6BBED388F44DEF8E34F777E787DA4F60
    40 OPERATIONS     BOSTON        BB0ED5A524D4D0DCBFA0F9CE98A3ED55
```

SQL Macros



SQL Macros



SQL Macros

```
select e.*,
       dbms_crypto.hash(src => utl_i18n.string_to_raw(to_char(EMPNO) || to_char(ENAME)
                                                      ,'AL32UTF8')
                        ,typ => 2) emphash
  from emp e
/
```

SQL Macros

```
select d.*,
       dbms_crypto.hash(src => utl_i18n.string_to_raw(to_char(DEPTNO) || to_char(DNAME) ||
                                                      to_char(LOC)
                                                      ,'AL32UTF8')
                        ,typ => 2) depthash
  from dept d
/
```

SQL Macros

```
select d.*,
       dbms_crypto.hash(src => utl_i18n.string_to_raw(to_char(DEPTNO) || to_char(DNAME) ||
                                                      to_char(LOC)
                                                      ,'AL32UTF8')
                        ,typ => 2) depthash
  from dept d
/
```

SQL Macros

```
create or replace function addhash(table_in dbms_tf.table_t) return varchar2 sql_macro is
  l_sql        varchar2(32767);
  l_hash       varchar2(32767);
begin
  for colidx in table_in.column.first .. table_in.column.last loop
    case -- check the datatype
      when table_in.column(colidx).description.type in
        (dbms_tf.type_varchar2, dbms_tf.type_number, dbms_tf.type_date) then
        l_hash := l_hash || q'[||to_char(]' || table_in.column(colidx).description.name || q'[)]';
      else
        null; --unsupported
    end case;
  end loop;
  l_hash := trim(both '|' from l_hash);
  l_hash := q'[dbms_crypto.hash(src => utl_i18n.string_to_raw(]' || l_hash ||
            q'[, 'AL32UTF8'), typ => ]' || dbms_crypto.hash_md5 || q'[) HashValue]';
  l_sql  := q'[select t.*, ]' || l_hash || q'[ from table_in t]';
  return l_sql;
end;
```

SQL Macros

```
select deptno, dname, loc, depthash
  from addhash(dept)
/

DEPTNO DNAME          LOC            DEPTHASH
------ -------------- ------------- --------------------------------
    10 ACCOUNTING     NEW YORK      9E8B838F90DC2CD43F345B7BF71A4313
    20 RESEARCH       DALLAS        782721AACDFDDCCFD4F834CB47B1C9C9
    30 SALES          CHICAGO       61EC3223E65F650824B77A1902D37A9D
    40 OPERATIONS     BOSTON        697109032E9C93E4BEACD1816058AE64
```

SQL Macros

```
create or replace function addhash(table_in   dbms_tf.table_t
                                  ,columns_in in dbms_tf.columns_t default null) return varchar2
sql_macro is
  l_sql          varchar2(32767);
  l_hash         varchar2(32767);
  l_columnname   varchar2(256);
  addthiscolumn boolean;

  function add_column(description_in in dbms_tf.column_metadata_t) return varchar2 is
    l_returnvalue varchar2(4000);
  begin
    case -- check the datatype
      when description_in.type in (dbms_tf.type_varchar2
                                  ,dbms_tf.type_number
                                  ,dbms_tf.type_date
                                  ,dbms_tf.type_raw
                                  ,dbms_tf.type_rowid
                                  ,dbms_tf.type_char
                                  ,dbms_tf.type_binary_float
                                  ,dbms_tf.type_binary_double
```

SQL Macros

```
    for colidx in table_in.column.first .. table_in.column.last loop
      l_hash := l_hash || add_column(table_in.column(colidx).description);
    end loop;
  else
    -- hash the columns provided (in that order)
    for cols in columns_in.first .. columns_in.last loop
      for colidx in table_in.column.first .. table_in.column.last loop
        if columns_in(cols) = table_in.column(colidx).description.name then
          l_hash := l_hash || add_column(table_in.column(colidx).description);
        end if;
      end loop;
    end loop;
  end if;
  l_hash := trim(both '|' from l_hash);
  l_hash := q'[dbms_crypto.hash(src => utl_i18n.string_to_raw(]' || l_hash || q'[, 'AL32UTF8'),
                typ => ]' || dbms_crypto.hash_md5 || q'[) ]' || l_columnname; -- || q'[hash]';
  l_sql  := q'[select t.*, ]' || l_hash || q'[ from table_in t]';
  return l_sql;
end;
/
```

SQL Macros

```
select deptno, dname, loc, depthash
  from addhash(dept, columns(deptno, dname))
/

DEPTNO DNAME          LOC            DEPTHASH
------ -------------- ------------- --------------------------------
    10 ACCOUNTING     NEW YORK      66C72E4D79A2D0602068405842C26ACD
    20 RESEARCH       DALLAS        EF0ADBB5676C17ACEA636C1D1A34B1FB
    30 SALES          CHICAGO       4EE0A666796B0829E60A3C297F20085F
    40 OPERATIONS     BOSTON        CA7B709739D67FDDED4DB3058328C422

select deptno, dname, loc, depthash
  from addhash(dept, columns(dname, deptno))
/

DEPTNO DNAME          LOC            DEPTHASH
------ -------------- ------------- --------------------------------
    10 ACCOUNTING     NEW YORK      BC0204E282FF670D9E265B8B178A5CAC
    20 RESEARCH       DALLAS        FC064F415267BC6C4D9FC98D3ACEC6AE
    30 SALES          CHICAGO       B98883E1727FF43F50F35E94026D4C28
    40 OPERATIONS     BOSTON        0084C3B8FE30880F28377361F639CBC1
```


Data integrity/quality
Integrity


Data integrity/quality
Integrity


Flashback queries


Data integrity/quality
Integrity


Flashback queries


Temporal validity


Data integrity/quality
Integrity


Flashback queries


Temporal validity


Data security


Data integrity/quality
Integrity


Flashback queries


Temporal validity


Data security


Performance increase
Results


Data integrity/quality
Integrity


Flashback queries


Temporal validity


Data security


Performance increase
Results


Special SQL features


Data integrity/quality
Integrity

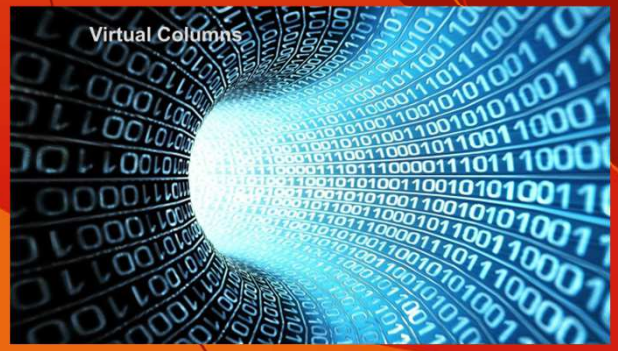

Flashback queries


Temporal validity


Data security


Performance increase
Results


Special SQL features


Virtual Columns


Data integrity/quality
Integrity

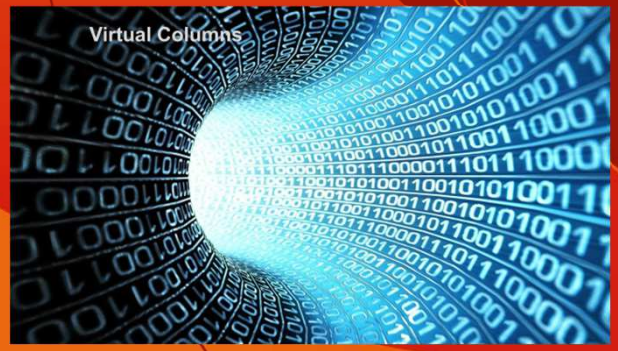

Flashback queries


Temporal validity


Data security


Performance increase
Results


Special SQL features


Virtual Columns


Identity columns (12c)
True Memories
of
The Good Old Days
QUALOGY


Data integrity/quality
Integrity

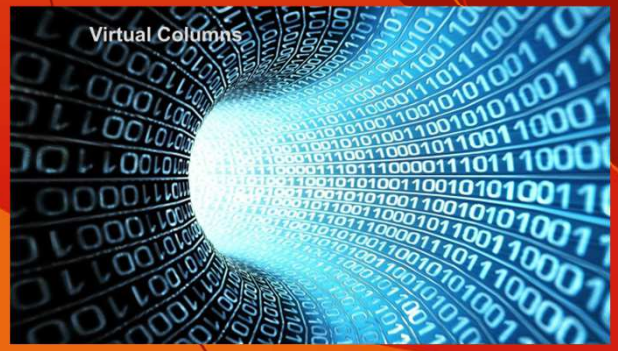

Flashback queries


Temporal validity


Data security


Performance increase
Results


Special SQL features


Virtual Columns

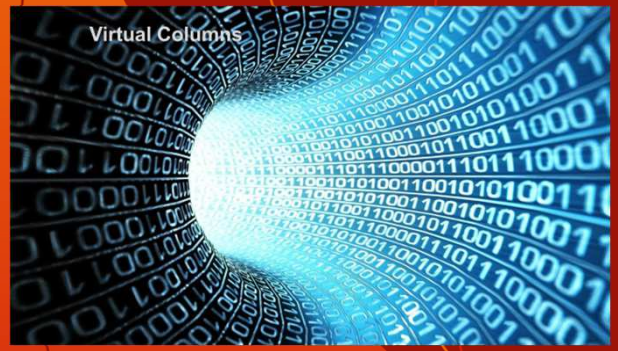

Identity columns (12c)
True Memories
of
The Good Old Days
QUALOGY


Invisible columns (12c)


Data security


Performance increase
Results


Special SQL features


Virtual Columns


Identity columns (12c)
True Memories
of
The Good Old Days
QUALOGY

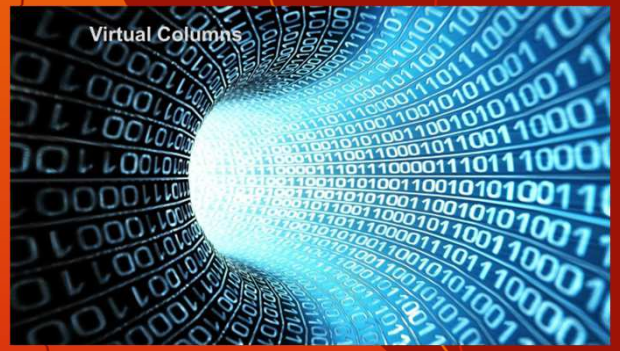

Invisible columns (12c)


Whitelisting PL/SQL program units
What the heck is "WHITELISTING?"
QUALOGY


Data security


Performance increase
Results

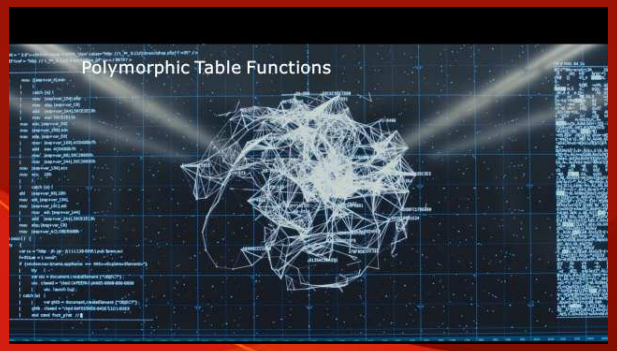

Special SQL features


Virtual Columns


Identity columns (12c)
True Memories
of
The Good Old Days
QUALOGY


Invisible columns (12c)

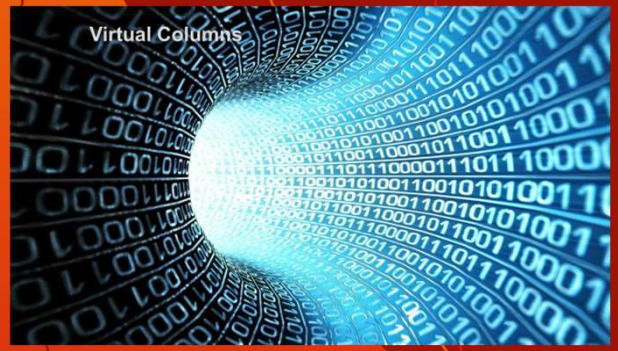

Whitelisting PL/SQL program units
What the heck is
"WHITELISTING?"
QUALOGY


Polymorphic Table Functions


Data security


Performance increase
Results

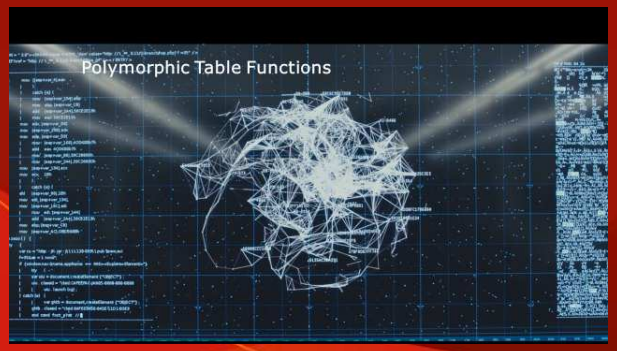

Special SQL features

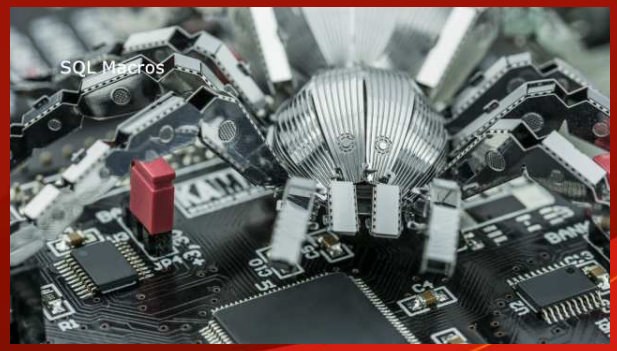

Virtual Columns


Identity columns (12c)
True Memories
of
The Good Old Days
QUALOGY


Invisible columns (12c)


Whitelisting PL/SQL program units
What the heck is
"WHITELISTING?"
QUALOGY


Polymorphic Table Functions


SQL Macros


Q&A

Oracle Cloud Infrastructure

New Free Tier

oracle.com/cloud/free

Always Free

Services you can use for unlimited time

+

30-Day Free Trial

Free credits you can use for more services