**SQL Oefeningen**

H1 – Standard Oefeningen

Part 1

1. True
2. True
3. Sal is an alias not an actual header, x has to be \*, last name has to have a comma behind it, ANNUAL SALARY has to be either small or in quotations. Correct statement:  
   SELECT employee\_id, last\_name, salary\*12 AS "Annual Salary"

FROM employees;

Part 2

1. SELECT employee\_id, last\_name, job\_id, hire\_date AS "STARTDATE"  
   FROM employees;
2. SELECT DISTINCT job\_id FROM employees;

Part 3

1. SELECT employee\_id AS "Emp#", last\_name AS "Employee", job\_id AS "Job", hire\_date AS "Hire Date"

FROM employees;

1. SELECT last\_name || ', ' || job\_id

AS "Employee and Title"

FROM employees

ORDER BY last\_name;

1. SELECT employee\_id || ', ' || first\_name || ', ' || last\_name || ', ' ||

email || ', ' || phone\_number || ', ' || job\_id || ', ' || hire\_date || ', ' ||

salary || ', ' || department\_id

AS "THE\_OUTPUT"

FROM employees;

H1 – Extra Oefeningen

1. SELECT \* FROM jobs;
2. SELECT \* FROM job\_history;
3. SELECT \* FROM departments;
4. DESCRIBE jobs; DESCRIBE job\_history; DESCRIBE departments;
5. SELECT department\_name, manager\_id FROM departments;
6. SELECT job\_id, max\_salary-min\_salary AS Verschil\_in\_salaris FROM jobs;
7. SELECT job\_id || ‘ is a ‘|| job\_title

AS functie

FROM jobs;

H2 – Standard Oefeningen

Part 1

1. SELECT last\_name, salary

FROM employees

WHERE salary > 12000;

1. SELECT last\_name, department\_id

FROM employees

WHERE last\_name = 'Taylor';

1. SELECT last\_name, salary

FROM employees

WHERE salary NOT BETWEEN 5000 AND 12000

1. SELECT last\_name, job\_id, hire\_date

FROM employees

WHERE last\_name = 'Matos'

OR last\_name = 'Taylor'

ORDER BY hire\_date;

1. SELECT last\_name, department\_id

FROM employees

WHERE department\_id = 20

OR department\_id = 50

ORDER BY last\_name;

1. SELECT last\_name AS "Employee", salary AS "Monthly Salary", department\_id

FROM employees

WHERE (department\_id = 50

OR department\_id = 20)

AND salary BETWEEN 5000 AND 12000;

1. SELECT last\_name, hire\_date

FROM employees

WHERE hire\_date LIKE '%94%';

1. SELECT last\_name, job\_id

FROM employees

WHERE manager\_id IS NULL;

1. SELECT last\_name, salary, commission\_pct

FROM employees

WHERE commission\_pct IS NOT NULL

ORDER BY salary, commission\_pct DESC;

1. SELECT last\_name, salary

FROM employees

WHERE salary > &employee\_salary;

1. SELECT employee\_id, last\_name, salary, department\_id

FROM employees

WHERE manager\_id = &manager\_id;

1. SELECT last\_name

FROM employees

WHERE last\_name LIKE '\_\_a%';

1. SELECT last\_name

FROM employees

WHERE last\_name LIKE'%a%'

AND last\_name LIKE '%e%';

1. SELECT last\_name, job\_id, salary

FROM employees

WHERE job\_id IN ('ST\_CLERK','SA\_REP')

AND (salary != 2500 AND salary != 3500 AND salary != 7000)

ORDER BY salary DESC;

1. SELECT last\_name AS "Employee", salary AS "Monthly Salary", commission\_pct

FROM employees

WHERE commission\_pct = 0.2;

Extra Oefeningen

1. SELECT employee\_id, last\_name, job\_id, hire\_date AS "STARTDATE"

FROM employees

WHERE (department\_id = 50 OR department\_id = 80 OR department\_id = 90);

1. SELECT employee\_id, last\_name, job\_id, hire\_date AS "STARTDATE"

FROM employees

WHERE manager\_id >= 200;

1. SELECT employee\_id, last\_name, job\_id, hire\_date AS "STARTDATE"

FROM employees

WHERE manager\_id IS NULL;

1. SELECT employee\_id, start\_date, end\_date

FROM job\_history

WHERE start\_date = '01-JAN-1999'

1. SELECT employee\_id, start\_date, end\_date

FROM job\_history

WHERE start\_date LIKE '%98'

AND department\_id = 50;

1. SELECT employee\_id, last\_name, job\_id, hire\_date AS "STARTDATE"

FROM employees

WHERE salary BETWEEN 5000 AND 10000

ORDER BY salary;

1. SELECT DISTINCT job\_id

FROM employees

WHERE (job\_id LIKE '%MAN%'

AND job\_id LIKE '\_A%');

1. SELECT DISTINCT job\_id

FROM employees

WHERE UPPER(job\_id) LIKE UPPER('%man%')

AND UPPER(job\_id) LIKE UPPER('\_a%');

1. SELECT \* FROM job\_history

WHERE department\_id = &employee\_department;

1. SELECT first\_name, last\_name

FROM employees

WHERE hire\_date < ('11/DEC/2001');

1. SELECT last\_name, department\_id, job\_id

FROM employees

WHERE department\_id != 80

AND (job\_id NOT LIKE 'ST\_CLERK' AND department\_id != 50);

1. SELECT last\_name, first\_name

FROM employees

WHERE last\_name LIKE 'D\_%'

OR last\_name LIKE'E\_%'

OR last\_name LIKE'F\_%'

OR last\_name LIKE'G\_%'

OR last\_name LIKE'H\_%'

OR last\_name LIKE'I\_%'

OR last\_name LIKE'J\_%';

1. SELECT job\_id

FROM employees;

1. SELECT first\_name, last\_name, salary, salary+salary\*0.1 AS "Toekomstige salaris"

FROM employees;

H3 – Standard Oefeningen

1. SELECT sysdate AS "Date"

FROM dual;

1. SELECT employee\_id, last\_name, salary, ROUND(salary+salary\*0.155) AS "New Salary"

FROM employees;

1. Done
2. SELECT employee\_id, last\_name, salary, ROUND(salary+salary\*0.155) AS "New Salary",

ROUND(salary+salary\*0.155)-salary AS "Increase"

FROM employees;

1. SELECT INITCAP(last\_name) AS "Name", LENGTH (last\_name) AS "Length"

FROM employees

WHERE last\_name LIKE 'J%'

OR last\_name LIKE 'A%'

OR last\_name LIKE 'M%';

SELECT INITCAP(last\_name) AS "Name", LENGTH (last\_name) AS "Length"

FROM employees

WHERE last\_name LIKE '&last\_name%';

SELECT INITCAP(last\_name) AS "Name", LENGTH (last\_name) AS "Length"

FROM employees

WHERE last\_name LIKE UPPER('&last\_name%');

1. SELECT last\_name, ROUND(MONTHS\_BETWEEN(sysdate, hire\_date), 0) AS MONTHS\_WORKED

FROM employees;

1. SELECT last\_name, LPAD(salary,15,'$')

FROM employees;

1. SELECT CONCAT(SUBSTR(last\_name,1,8), RPAD(' \*',round(salary/1000),'\*'))

AS "EMPLOYEES\_AND\_THEIR\_SALARIES"

FROM employees

ORDER BY salary DESC;

1. SELECT last\_name, TRUNC((sysdate-hire\_date)/7,0) AS TENURE  
   FROM employees  
   WHERE department\_id = 90  
   ORDER BY TRUNC((sysdate-hire\_date)/7,0) DESC

Extra Oefeningen

1. SELECT RPAD('#',round(min\_salary/1000),'#')

AS "MIN\_SALARY"

FROM jobs;

1. SELECT CONCAT(SUBSTR(job\_id,1,2), SUBSTR(job\_id,4,10)) AS "SUBSTR", job\_title

FROM jobs

WHERE max\_salary < 10000

1. SELECT REPLACE(job\_id, '\_', ' ') AS "SUBSTR", job\_title

FROM jobs

WHERE max\_salary < 10000;

1. Foute referentie naar oefening?
2. SELECT ROUND(MONTHS\_BETWEEN(end\_date, start\_date)) AS "MONTHS\_WORKED",

LAST\_DAY(end\_date) AS "LAST\_DAY"

FROM job\_history

WHERE ROUND(MONTHS\_BETWEEN(end\_date, start\_date)) > 12;

H4 – Standard Oefeningen

1. SELECT last\_name || ' earns' || TO\_CHAR(salary, '$99,999.00')

|| ' monthly but wants ' || TO\_CHAR(salary\*3, '$99,999.00') AS "Dream Salaries"

FROM employees;

1. SELECT last\_name, hire\_date,

TO\_CHAR(NEXT\_DAY(ADD\_MONTHS(hire\_date,6), 'MONDAY'),'fmDay, Ddspth "of" Month, YYYY') AS "REVIEW"

FROM employees;

SELECT last\_name, hire\_date,

TO\_CHAR(hire\_date,'fmDAY') AS "DAY"

FROM employees

ORDER BY TO\_CHAR(hire\_date,'fmDAY');

1. SELECT last\_name, NVL(TO\_CHAR(commission\_pct), 'No Commission') "COMM"

FROM employees;

SELECT job\_id,

DECODE(job\_id,

'AD\_PRES', 'A',

'ST\_MAN', 'B',

'IT\_PROG', 'C',

'SA\_REP', 'D',

'ST\_CLERK', 'E',

'0') "Grade"

FROM employees;

SELECT job\_id,   
CASE job\_id   
WHEN 'AD\_PRES' THEN 'A'  
WHEN 'ST\_MAN' THEN 'B'  
WHEN 'IT\_PROG' THEN 'C'  
WHEN 'SA\_REP' THEN 'D'  
WHEN 'ST\_CLERK' THEN 'E'  
ELSE '0' END "Grade  
FROM employees;

Extra Oefeningen

1. SELECT last\_name,

TO\_CHAR(NEXT\_DAY(hire\_date, 'Friday'),'fmDay, "1st of November 2017"')

AS "AANWERVINGSDATUM",

last\_name || ' werkt al ' ||

ROUND(MONTHS\_BETWEEN(sysdate, hire\_date)/12) || ' jaren bij ons'

AS "Dienstjaren"

FROM employees;

1. SELECT last\_name, salary, commission\_pct,

NVL(TO\_CHAR(commission\_pct\*salary), 'Geen Kommissie') "MONTHLY COMM"

FROM employees

ORDER BY NVL(TO\_CHAR(commission\_pct\*salary), 'Geen Kommissie');

1. SELECT last\_name, job\_id, salary,

CASE WHEN salary<5000 THEN salary\*0.15+salary

WHEN salary<10000 THEN salary\*0.10+salary

ELSE salary END "NIEUW SALARIS"

FROM employees;

H6 – Standard Oefeningen

1. SELECT l.location\_id, l.street\_address, l.city, l.state\_province,

c.country\_name

FROM locations l JOIN countries c

ON (l.country\_id = c.country\_id);

1. SELECT e.last\_name, e.department\_id, d.department\_name

FROM employees e JOIN departments d

ON (e.department\_id = d.department\_id)

ORDER BY department\_id;

1. SELECT e.last\_name, e.job\_id, e.department\_id, d.department\_name

FROM employees e JOIN departments d

ON (d.department\_id = e.department\_id)

JOIN locations l

ON d.location\_id = l.location\_id

WHERE l.city = 'Toronto';

1. SELECT worker.last\_name "Employee", worker.job\_id "EMP#",

manager.last\_name "Manager", manager.job\_id "Mgr#"

FROM employees worker JOIN employees manager

ON (worker.manager\_id = manager.employee\_id);

1. SELECT worker.last\_name "Employee", worker.employee\_id "EMP#",

manager.last\_name "Manager", manager.employee\_id "Mgr#"

FROM employees worker LEFT OUTER JOIN employees manager

ON (worker.manager\_id = manager.employee\_id)

ORDER BY worker.employee\_id;

6. SELECT e.department\_id "Department",

e.last\_name "Employee",

c.last\_name "Colleague"

FROM employees e JOIN employees c

ON (e.department\_id = c.department\_id)

WHERE e.employee\_id <> c.employee\_id

ORDER BY e.department\_id

1. SELECT e.last\_name, e.job\_id, d.department\_name, e.salary, g.grade\_level

FROM employees e JOIN departments d

ON (e.department\_id = d.department\_id)

JOIN job\_grades g

ON (e.salary BETWEEN g.lowest\_sal AND g.highest\_sal);

1. SELECT last\_name, hire\_date

FROM employees

WHERE hire\_date > '29-JAN-97';

OR

SELECT e.last\_name, e.hire\_date

FROM employees e JOIN employees davies

ON (davies.last\_name = 'Davies')

WHERE davies.hire\_date < e.hire\_date;

1. SELECT e.last\_name, e.hire\_date, m.last\_name, m.hire\_date

FROM employees e JOIN employees m

ON (e.manager\_id = m.employee\_id)

WHERE e.hire\_date < m.hire\_date;

Extra Oefeningen

1. SELECT h.employee\_id, e.last\_name, e.job\_id, j.job\_title

FROM employees e RIGHT OUTER JOIN job\_history h

ON (h.employee\_id = e.employee\_id)

LEFT OUTER JOIN jobs j

ON (j.job\_id = h.job\_id);

1. SELECT h.employee\_id, e.last\_name, e.job\_id, j.job\_title

FROM employees e FULL OUTER JOIN job\_history h

ON (h.employee\_id = e.employee\_id)

JOIN jobs j

ON (j.job\_id = h.job\_id)

WHERE

UPPER(e.job\_id) LIKE 'AD\_%'

OR UPPER(e.job\_id) LIKE 'SA\_%';

1. SELECT h.employee\_id, e.last\_name, e.job\_id, j.job\_title

FROM employees e JOIN job\_history h

ON (h.employee\_id = e.employee\_id)

JOIN jobs j

ON (j.job\_id = h.job\_id)

JOIN employees abel

ON (abel.job\_id = h.job\_id)

WHERE INITCAP(abel.last\_name) = 'Abel'

AND INITCAP(abel.first\_name) = ‘Ellen’;

1. SELECT e.department\_id, d.department\_name,

e.first\_name ||' '|| e.last\_name "Afdelingshoofd"

FROM departments d JOIN employees e

ON (d.manager\_id = e.employee\_id)

WHERE d.department\_id = 90;

1. SELECT UPPER(e.first\_name ||' '|| e.last\_name) "Afdelingshoofd"

FROM employees e JOIN departments d

ON (e.department\_id = d.department\_id)

WHERE e.manager\_id != d.manager\_id;

1. Select l.\*, c.country\_name, r.region\_name

FROM locations l JOIN countries c

ON (l.country\_id = c.country\_id);

JOIN regions r

ON (c.region\_id = r.region\_id);

1. SELECT e.last\_name, e.hire\_date, c.country\_name

FROM employees e JOIN departments d

ON (e.department\_id = d.department\_id)

JOIN locations l

ON (d.location\_id = l.location\_id)

JOIN countries c

ON (l.country\_id = c.country\_id)

WHERE c.country\_name = 'United States of America'

AND e.hire\_date > &hire\_date; ’01-JAN-1995’

1. Kennis hoofdstuk 5 nodig blijkbaar
2. SELECT e.employee\_id, e.last\_name, e.manager\_id, h.start\_date, h.end\_date, e.job\_id

FROM employees e LEFT OUTER JOIN job\_history h

ON (e.employee\_id = h.employee\_id)

WHERE e.department\_id != 80 AND

e.department\_id != 90

OR e.department\_id IS NULL;

1. SELECT e.last\_name, e.manager\_id, e.department\_id, d.manager\_id "department man"

FROM employees e JOIN departments d

ON (e.department\_id = d.department\_id)

WHERE (d.manager\_id = e.manager\_id);

H9 – Standard Oefeningen

1. done
2. done
3. Using Implicit method:

INSERT INTO my\_employee VALUES (1, 'Patel', 'Ralph', 'rpatel', 895);

1. Explicit:

INSERT INTO my\_employee (id, last\_name, first\_name, userid, salary)

VALUES (2, 'Dancs', 'Betty', 'bdancs', 860);

1. SELECT \* FROM my\_employee;
2. INSERT INTO my\_employee   
   VALUES (&p\_id, '&p\_last\_name', '&p\_first\_name', '&p\_userid', &p\_salary);
3. done
4. SELECT \* FROM my\_employee;
5. COMMIT;
6. UPDATE my\_employee

SET last\_name = 'Drexler'

WHERE last\_name = 'Biri';

1. UPDATE my\_employee

SET salary = 1000

WHERE salary < 900;

1. SELECT \* FROM my\_employee;
2. DELETE FROM my\_employee

WHERE last\_name = 'Dancs';

1. SELECT \* FROM my\_employee;
2. COMMIT;
3. done
4. yes
5. SAVEPOINT A;
6. DELETE FROM my\_employee;
7. yes
8. ROLLBACK TO A;
9. yes
10. COMMIT;
11. INSERT INTO my\_employee

VALUES (&p\_id, '&&p\_last\_name', '&&p\_first\_name',

lower(substr('&p\_first\_name', 1, 1) ||

substr('&p\_last\_name', 1, 7)), &p\_salary);

1. done
2. SELECT \*   
   FROM my\_employee   
   WHERE ID='6';

H10 – Standard Oefeningen

1. CREATE TABLE dept

(id NUMBER(7)CONSTRAINT department\_id\_pk PRIMARY KEY,

name VARCHAR2(25));

DESC DEPT;

1. INSERT INTO dept

SELECT department\_id, department\_name

FROM departments;

1. CREATE TABLE employees2 AS

SELECT employee\_id id, first\_name, last\_name, salary, department\_id dept\_id

FROM employees;

1. INSERT INTO employees2

(id, first\_name, last\_name, salary, dept\_id)

VALUES (34, 'Grant', 'Marcie', '5678', 10);

1. ALTER TABLE employees2 READ ONLY;
2. INSERT INTO employees2

(id, first\_name, last\_name, salary, dept\_id)

VALUES (34, 'Grant', 'Marcie', '5678', 10);

1. ALTER TABLE employees READ WRITE

INSERT INTO employees2

(id, first\_name, last\_name, salary, dept\_id)

VALUES (34, 'Grant', 'Marcie', '5678', 10);

1. DROP TABLE employees2;

Extra Oefeningen H9/10

1. REM ===============

REM TABEL CURSUSSEN

REM ===============

DROP TABLE cursussen CASCADE CONSTRAINTS;

CREATE TABLE cursussen (

curs\_code VARCHAR2(4) CONSTRAINT cursussen\_curs\_code\_pk PRIMARY KEY

, curs\_omschr VARCHAR2(20) CONSTRAINT cursussen\_curs\_omschr\_nn NOT NULL

, curs\_type VARCHAR2(10) CONSTRAINT cursussen\_curs\_type\_nn NOT NULL

CONSTRAINT cursussen\_curs\_type\_ck CHECK (CURS\_TYPE = UPPER(CURS\_TYPE))

, curs\_lengte NUMBER(2) CONSTRAINT cursussen\_curs\_lengte CHECK (CURS\_LENGTE BETWEEN 1 AND 10)

);

REM ==================

REM TABEL UITVOERINGEN

REM ==================

DROP TABLE uitvoeringen CASCADE CONSTRAINTS;

CREATE TABLE uitvoeringen (

cursus VARCHAR2(4) CONSTRAINT uitv\_cursus\_nn NOT NULL

CONSTRAINT uitv\_cursus\_fk REFERENCES cursussen(curs\_code)

, begindatum DATE CONSTRAINT uitv\_begindatum\_nn NOT NULL

, docent NUMBER(4) CONSTRAINT uitv\_docent\_fk REFERENCES employees(employee\_id)

, locatie VARCHAR2(20)

, CONSTRAINT uitv\_cursus\_begindatum\_pk PRIMARY KEY(cursus, begindatum)

);

REM ====================

REM TABEL INSCHRIJVINGEN

REM ====================

DROP TABLE inschrijvingen CASCADE CONSTRAINTS;

CREATE TABLE inschrijvingen (

cursist NUMBER(4) CONSTRAINT inschr\_cursist\_nn NOT NULL

CONSTRAINT inschr\_cursist\_fk REFERENCES employees(employee\_id)

, cursus VARCHAR2(4) CONSTRAINT inschr\_cursus\_nn NOT NULL

, begindatum DATE CONSTRAINT inschr\_begindatum\_nn NOT NULL

, evaluatie NUMBER(1) CONSTRAINT inschr\_evaluatie\_ck CHECK (evaluatie BETWEEN 1 AND 5)

, CONSTRAINT inschr\_cursist\_cursus\_pk PRIMARY KEY(cursist, cursus)

, CONSTRAINT inschr\_cursus\_begindatum\_fk FOREIGN KEY(cursus, begindatum) REFERENCES uitvoeringen(cursus, begindatum)

);

1. INSERT INTO cursussen (curs\_code, curs\_omschr, curs\_type, curs\_lengte)

VALUES ('&curs\_code', '&curs\_omschr', '&curs\_type', &curs\_lengte)

/

INSERT INTO uitvoeringen (cursus, begindatum, docent, locatie)

VALUES (‘&curs\_code’, '&datum', &docent, '&locatie')

/

INSERT INTO inschrijvingen(cursist, cursus, begindatum)

VALUES (&cursist, '&cursus', '&datum')

/

1. UPDATE uitvoeringen

SET locatie = 'Leuven'

WHERE cursus = 'SQL2' AND begindatum = '04-JUN-2014'

/

1. /
2. /
3. DELETE from uitvoeringen

WHERE locatie = 'Hasselt'

/

Error message: integrity constraint (STUDENT.INSCHR\_CURSUS\_BEGINDATUM\_FK) violated - child record found

1. ROLLBACK;
2. CREATE TABLE curs\_history AS SELECT \* FROM cursussen;
3. DROP TABLE curs\_history;
4. DROP TABLE King\_medewerkers CASCADE CONSTRAINTS;

CREATE TABLE King\_medewerkers

AS SELECT employee\_id, last\_name, first\_name, department\_id, TRUNC(SYSDATE, 'MONTH') AS registratie

FROM employees

WHERE manager\_id = 100

/

1. DELETE FROM King\_medewerkers

WHERE last\_name LIKE '% %'

/

1. /
2. /
3. /

Coloumn level:

1. REM ===============

REM TABEL ARTIKELS

REM ===============

DROP TABLE artikels CASCADE CONSTRAINTS;

CREATE TABLE artikels (

id VARCHAR2(10) CONSTRAINT artikels\_id\_pk PRIMARY KEY

, naam VARCHAR2(30) CONSTRAINT artikels\_curs\_omschr\_nn NOT NULL

, prijs DECIMAL(8,2)

, datum\_van\_bestaan DATE

, land\_fabricatie CHAR(2) CONSTRAINT artikels\_land\_fk REFERENCES countries(country\_id)

);

Table level:

REM ===============

REM TABEL ARTIKELS

REM ===============

DROP TABLE artikels CASCADE CONSTRAINTS;

CREATE TABLE artikels (

id VARCHAR2(10)

, naam VARCHAR2(30)

, prijs DECIMAL(8,2)

, datum\_van\_bestaan DATE

, land\_fabricatie CHAR(2)

, CONSTRAINT artikels\_id\_pk PRIMARY KEY (id)

, CONSTRAINT artikels\_land\_fk FOREIGN KEY (land\_fabricatie)

REFERENCES countries (country\_id)

);

1. REM ===============

REM TABEL MAGAZIJNEN

REM ===============

DROP TABLE magazijnen CASCADE CONSTRAINTS;

CREATE TABLE magazijnen (

magazijn\_nummer NUMBER(20) CONSTRAINT magazijn\_nummer\_pk PRIMARY KEY

, naam\_magazijn VARCHAR2(30) CONSTRAINT magazijn\_naam\_nn NOT NULL

, staat VARCHAR2(30) CONSTRAINT magazijn\_straat\_nn NOT NULL

, huisnummer NUMBER(4) CONSTRAINT magazijn\_huisnummer\_nn NOT NULL

, gemeente VARCHAR2(30) CONSTRAINT magazijn\_gemeente\_nn NOT NULL

);

REM =======================

REM TABEL MAGAZIJN\_ARTIKELS

REM =======================

DROP TABLE magazijn\_artikels CASCADE CONSTRAINTS;

CREATE TABLE magazijn\_artikels (

magazijn\_nummer NUMBER(20) CONSTRAINT magazijn\_nummer\_fk

REFERENCES magazijnen(magazijn\_nummer)

, artikel\_nummer NUMBER(30) CONSTRAINT artikel\_nummer\_nn NOT NULL

, aantal\_artikels NUMBER(10)

, CONSTRAINT magazijn\_num\_art\_pk PRIMARY KEY(magazijn\_nummer, artikel\_nummer)

)

/

1. A) REM ===============

REM TABEL SPELERS

REM ===============

DROP TABLE spelers CASCADE CONSTRAINTS;

CREATE TABLE spelers (

spelers\_nummer NUMBER(5) CONSTRAINT spelers\_nummer\_pk PRIMARY KEY

, spelers\_naam VARCHAR2(20) CONSTRAINT spelers\_naam\_nn NOT NULL

, speler\_voorletter VARCHAR2(3)

, geslacht CHAR(1) DEFAULT 'M' CONSTRAINT spelers\_geslacht\_nn NOT NULL

, ploeg\_nr NUMBER(3)

);

REM =======================

REM TABEL TEAMS

REM =======================

DROP TABLE teams CASCADE CONSTRAINTS;

CREATE TABLE teams (

team\_nr NUMBER(3) CONSTRAINT team\_nr\_pk PRIMARY KEY

, team\_naam VARCHAR2(20) CONSTRAINT team\_naam\_nn NOT NULL

CONSTRAINT team\_naam\_ck CHECK (TEAM\_NAAM = UPPER(TEAM\_NAAM))

, kapitein NUMBER(5) CONSTRAINT team\_kapitein\_nn NOT NULL

);

REM ===============

REM TABEL BOETES

REM ===============

DROP TABLE boetes CASCADE CONSTRAINTS;

CREATE TABLE boetes (

boete\_nummer NUMBER(3) CONSTRAINT boete\_nummer\_pk PRIMARY KEY

, spelers\_nr NUMBER(5) CONSTRAINT spelers\_nr\_fk

REFERENCES spelers(spelers\_nummer)

, datum DATE CONSTRAINT datum\_nn NOT NULL

, bedrag DECIMAL(7,2) CONSTRAINT spelers\_bedrag\_nn NOT NULL

CHECK (bedrag > 100)

)

/

B)

C)

D)

E)

F)

G)

H11 – Standard Oefeningen

1. CREATE OR REPLACE VIEW EMPLOYEES\_VU

AS SELECT employee\_id, last\_name EMPLOYEE, department\_id

FROM employees;

1. SELECT \* FROM EMPLOYEES\_VU;
2. SELECT employee, department\_id FROM employees\_vu;
3. CREATE OR REPLACE VIEW DEPT50

AS SELECT employee\_id EMPNO, last\_name EMPLOYEE, department\_id DEPTNO

FROM employees

WHERE department\_id = 50

WITH CHECK OPTION CONSTRAINT emp\_dept\_50;

1. SELECT \* FROM dept50;
2. UPDATE dept50

SET deptno = 80

WHERE employee = 'Matos';

1. CREATE SEQUENCE dept\_id\_seq

START WITH 200

INCREMENT BY 10

MAXVALUE 1000;

1. INSERT INTO dept

VALUES (dept\_id\_seq.nextval, 'Education');

INSERT INTO dept

VALUES (dept\_id\_seq.nextval, 'Administration');

1. CREATE INDEX dept\_name\_idx

ON dept (name);

1. CREATE SYNONYM emp FOR EMPLOYEES;

Extra Oefeningen

1. CREATE OR REPLACE VIEW loc\_us\_vu

AS SELECT location\_id, street\_address, city, country\_id

FROM locations

WHERE country\_id = 'US';

1. CREATE OR REPLACE VIEW loc\_us\_vu

AS SELECT location\_id LOCATION, street\_address STREET, city CITY, country\_id COUNTRY

FROM locations

WHERE country\_id = 'US';

1. UPDATE loc\_us\_vu

SET city = ‘Seatle’

WHERE location = 1700;

1. CREATE OR REPLACE VIEW loc\_us\_vu

AS SELECT location\_id LOCATION, street\_address STREET, city CITY, country\_id COUNTRY

FROM locations

WHERE country\_id = 'US'

WITH CHECK OPTION CONSTRAINT country\_country\_id\_us;

1. UPDATE loc\_us\_vu

SET country = 'CA'

WHERE location = 1700; 🡪 Niet toegestaan

1. CREATE OR REPLACE VIEW job\_sal\_vu

AS SELECT e.last\_name

FROM employees e join employees m

ON (m.employee\_id = e.manager\_id)

WHERE e.salary > m.salary

WITH READ ONLY CONSTRAINT job\_sal\_vu\_RO

/

1. INSERT INTO employees   
   (job\_id, last\_name, hire\_date, department\_id)  
   VALUES (&job\_id, '&last\_name', TO\_DATE(‘&hire\_date’, ‘DD/MM/YYYY’) , &department\_id)  
   /

H5 – Standard Oefeningen

1. True
2. False
3. True
4. SELECT ROUND(MAX(salary),0) "Maximum",

ROUND(MIN(salary),0) "Minimum",

ROUND(SUM(salary),0) "Sum",

ROUND(AVG(salary),0) "Average"

FROM employees;

1. SELECT job\_id,

ROUND(MAX(salary),0) "Maximum",

ROUND(MIN(salary),0) "Minimum",

ROUND(SUM(salary),0) "Sum",

ROUND(AVG(salary),0) "Average"

FROM employees

GROUP BY job\_id;

1. SELECT job\_id, COUNT(employee\_id)

FROM employees

GROUP BY job\_id;

SELECT job\_id, COUNT(\*)

FROM employees

WHERE job\_id = '&job\_title'

GROUP BY job\_id;

1. SELECT COUNT(DISTINCT manager\_id) "Number of Managers"

FROM employees;

1. SELECT MAX(salary)-MIN(salary) DIFFERENCE

FROM employees;

1. SELECT manager\_id, MIN(salary)

FROM employees

WHERE manager\_id IS NOT NULL

GROUP BY manager\_id

HAVING MIN(salary) > 6000

ORDER BY MIN(salary) DESC;

1. SELECT COUNT(\*) total,

SUM(DECODE(TO\_CHAR(hire\_date, 'YYYY'),1995,1,0))"1995",

SUM(DECODE(TO\_CHAR(hire\_date, 'YYYY'),1996,1,0))"1996",

SUM(DECODE(TO\_CHAR(hire\_date, 'YYYY'),1997,1,0))"1997",

SUM(DECODE(TO\_CHAR(hire\_date, 'YYYY'),1998,1,0))"1998"

FROM employees;

1. SELECT job\_id "Job",

SUM(DECODE(department\_id , 20, salary)) "Dept 20",

SUM(DECODE(department\_id , 50, salary)) "Dept 50",

SUM(DECODE(department\_id , 80, salary)) "Dept 80",

SUM(DECODE(department\_id , 90, salary)) "Dept 90",

SUM(salary) "Total"

FROM employees GROUP BY job\_id;

Extra Oefeningen

1. SELECT manager\_id, COUNT(employee\_id)

FROM employees

WHERE manager\_id IS NOT NULL

GROUP BY manager\_id

1. SELECT manager\_id, COUNT(employee\_id)

FROM employees

WHERE manager\_id IS NOT NULL

GROUP BY manager\_id

HAVING MOD(manager\_id, 10) = 0;

1. SELECT manager\_id, COUNT(employee\_id)

FROM employees

WHERE manager\_id IS NOT NULL AND employee\_id >= 3

GROUP BY manager\_id

HAVING MOD(manager\_id, 10) = 0;

1. SELECT Country\_id, COUNT(\*) LOCATIONS

FROM locations

GROUP BY country\_id;

1. /
2. /
3. /
4. /
5. /
6. /
7. /

H7 – Standard Oefeningen

1. SELECT last\_name, hire\_date

FROM employees

WHERE department\_id =

(Select department\_id

FROM employees

WHERE last\_name = '&&Enter\_name')

AND last\_name <> '&&Enter\_name'

/

UNDEFINE Enter\_name

(save in script file, otherwise undef doesn’t work)

1. SELECT employee\_id, last\_name, salary

FROM employees

WHERE salary >

(Select AVG(salary)

FROM employees)

ORDER BY salary

/

1. SELECT employee\_id, last\_name

FROM employees

WHERE department\_id IN (SELECT department\_id

FROM employees

WHERE last\_name like '%u%')

/

1. SELECT last\_name, department\_id, job\_id

FROM employees

WHERE department\_id IN (SELECT department\_id

FROM departments

WHERE location\_id = 1700)

/

SELECT last\_name, department\_id, job\_id

FROM employees

WHERE department\_id IN (SELECT department\_id

FROM departments

WHERE location\_id = &location\_id)

/

1. SELECT last\_name, salary

FROM employees

WHERE manager\_id = (SELECT employee\_id

FROM employees

WHERE UPPER(last\_name) = 'KING')

/

1. SELECT department\_id, last\_name, job\_id

FROM employees

WHERE department\_id = (SELECT department\_id

FROM departments

WHERE UPPER(department\_name) = 'EXECUTIVE')

/

1. SELECT employee\_id, last\_name, salary

FROM employees

WHERE department\_id IN (SELECT department\_id

FROM employees

WHERE last\_name like '%u%')

AND salary > (SELECT AVG(salary)

FROM employees)

/

Extra Oefeningen H7

SELECT employee\_id, last\_name, job\_id

FROM employees

WHERE job\_id = (SELECT job\_id

FROM employees

WHERE UPPER(last\_name) = 'ABEL')

AND UPPER(last\_name) <> 'ABEL'

/