- počet všetkých zamestnancov

```
SELECT count(*) FROM employees;
```

- počet oddelení, ktorí majú aspoň jedného zamestnanca

```
SELECT count(distinct department_id) FROM

employees;
```

- koľko zarábajú zamestnanci, pracujúci pre oddelenie Marketing (id oddelenia je 20)

```
SELECT sum(salary) FROM employees WHERE department_id=20;
```

- počet zamestnancov v jednotlivých oddeleniach

```
SELECT department_id, count(*)

FROM employees GROUP BY department_id

ORDER BY 1;
```

- minimálny plat zamestnancov v jednotlivých oddeleniach

```
SELECT department_id, min(salary)
FROM employees GROUP BY department_id

ORDER BY 1;
```

- zobrazenie počtu zamestnancov pre jednotlivé pozície a oddelenia

```
SELECT department_id,job_id,count(*)

FROM employees GROUP BY department_id,job_id

ORDER BY 1;
```

zobrazenie oddelení, v ktorých je priemerná mzda väčšia ako 5000

```
 SELECT department_id, avg(salary)
FROM employees
GROUP BY department_id HAVING avg(salary)>5000;
```

- zobrazenie oddelení, kde pracujú aspoň jeden obchodný zástupca

```
 SELECT department_id,job_id,count(*)
FROM employees WHERE job_id='SA_REP'
GROUP BY department_id,job_id HAVING count(*)>=1;
```

- najdlhšie pracujúci zamestnanec na aktuálnej pozícii

```
SELECT employee_id, first_name, last_name, salary FROM employees

WHERE salary=(SELECT max(salary) FROM employees);
```

- zobrazte oddelenie a ich min plat, ktorý je väčší ako maximálny plat v oddelení 10

```
   SELECT department_id, min(salary)
      FROM employees
GROUP BY department_id
HAVING min(salary)>(SELECT max(salary) FROM employees
```

WHERE department_id=10);

- zobraziť manažérov oddelení Human resources a Administration

SELECT employee_id, first_name, last_name, department_id FROM employees
WHERE employee_id
IN (SELECT manager_id FROM departments WHERE department_name IN ('Human Resources','Administration'));

zobrazte zamestnancov, ktorý zarábajú viac ako aspoň jeden zo zamestnancov oddelenia Administration

SELECT employee_id, first_name, last_name, salary FROM employees

WHERE salary>ANY (SELECT salary FROM employees WHERE department_id=(SELECT department_id FROM departments
WHERE Department_name='Administration'));

- zoznam zamestnancov, ktorí zarábajú menej ako akýkoľvek zamestnanec oddelenia Administration

SELECT employee_id, first_name, last_name, salary FROM employees

WHERE salary<ALL(SELECT salary FROM employees WHERE department_id=(SELECT department_id FROM
departments WHERE department_name='Administration'))

ORDER BY 4;

- **zoznam zamestnancov, ktorých plat je vyšší ako priemerný plat v oddelení, v ktorom pracuje**

SELECT first_name, last_name, salary, department_id FROM employees e1

WHERE salary>(SELECT avg(salary) FROM employees WHERE department_id=e1.department_id);

- **zoznam oddelení a počet zamestnancov**

SELECT department_name, (SELECT count(*) FROM employees e WHERE e.department_id=d.department_id) AS count_dep FROM
departments d;

zobrazte zoznam oddelení, ktoré majú/nemajú zamestnancov

SELECT department_name FROM departments d WHERE NOT EXISTS (SELECT 1 FROM employees e WHERE
e.department_id=d.department_id);

- zobrazenie id zamestnanca, ktorý pracoval vo firme na najviac

pracovných pozíciach

SELECT employee_id,count(*)+1
FROM job_history GROUP BY employee_id HAVING count(*)=(SELECT

max(Pocet) FROM (SELECT employee_id, count(*) POCET FROM job_history GROUP BY employee_id));

- zobrazit nazvy oddeleni a pocet zamestnancov pre jednotlive

oddelenia

SELECT department_name, (SELECT count(*) FROM employees e WHERE e.department_id=d.department_id) AS count_dep FROM
departments d;

- vytvorte tabuľku emp_dep, ktorá bude obsahovať id a meno zamestnanca,plat,meno a id oddelenia(<50) pomocou tabuliek employees a departments
CREATE TABLE emp_dep AS
(SELECT e.employee_id, e.first_name||' '||e.last_name name, e.salary, d.department_name, d.department_id FROM employees e LEFT
OUTER JOIN departments d ON e.department_id=d.department_id WHERE d.department_id<50);

- do tabuľky emp_dep vložte informácie o zamestnancoch oddelení s id >50

```
INSERT INTO emp_dep (SELECT e.employee_id, e.first_name||' '||e.last_name name, e.salary, d.department_name, d.department_id
FROM employees e LEFT OUTER JOIN departments d ON e.department_id=d.department_id WHERE d.department_id>=50);
```

- zmazať všetkých zamestnancov, ktorý zarabajú menej ako priemerný plat

```
DELETE FROM employees WHERE salary<(SELECT avg(salary)
  FROM employees);
```

- zmazať oddelenia, ktoré nemajú zamestnanca

```
DELETE FROM departments WHERE department_id IN
(SELECT department_id FROM departments d WHERE NOT EXISTS
  (SELECT 1 FROM employees e WHERE
  e.department_id=d.department_id));
```

zvýšiť plat o 100 všetkým zamestnancom, ktorý majú plat nižší ako priemer

```
UPDATE employees SET salary=salary+100 WHERE salary<(SELECT avg(salary) FROM

employees);
```

- nastaviť plat všetkým zamestnancom na hodnotu priemerného platu oddelenia, v ktorom pracujú

```
UPDATE employees e1 SET salary=(SELECT avg(salary) FROM employees e2 WHERE e2.department_id=e1.department_id);
```

```
CREATE OR REPLACE TRIGGER print_salary_changes
  BEFORE DELETE OR INSERT OR UPDATE ON employees
FOR EACH ROW

WHEN (NEW.job_id <> 'AD_PRES') -- do not print information about President

DECLARE
sal_diff NUMBER;

BEGIN

  sal_diff  := :NEW.salary  - :OLD.salary;
  DBMS_OUTPUT.PUT(:NEW.last_name || ': ');
DBMS_OUTPUT.PUT('Old salary = ' || :OLD.salary || ', ');

  DBMS_OUTPUT.PUT('New salary = ' || :NEW.salary || ',
  ');
DBMS_OUTPUT.PUT_LINE('Difference: ' || sal_diff); END;
```

```
CREATE USER janko IDENTIFIED BY hrasko321;

CREATE USER janko IDENTIFIED BY hrasko321 DEFAULT TABLESPACE users;

CREATE USER janko IDENTIFIED BY hrasko321 DEFAULT TABLESPACE users

ACCOUNT LOCK PASSWORD EXPIRED;
```

Prihlásenie:

OS: sqlplus <username>/<password> sqlplus: connect <username>/<password>

```
GRANT <privilege> TO <grantee> [WITH ADMIN OPTION];
```

REVOKE <privilege> FROM <grantee> ;

GRANT <privilege> ON <object name> TO <grantee> [WITH GRANT OPTION];

REVOKE <privilege> ON <object name> FROM
<grantee> ;

SQL dopyt, ktorý vráti počet features, ktoré majú ohlásený nejaký bug. Stĺpec s výsledkom pomenujte „BUGGED_FEAT“. Je ich 3.
SELECT COUNT(DISTINCT feature_id) AS "BUGGED_FEAT"
FROM bug_feature;

SQL dopyt, vráti celé záznamy komentárov, ktoré boli zverejnené v iný mesiac ako apríl (na roku nezáleží) zoradené podľa dátumu (od najstaršieho po najnovší). Majú id 7 a 10.
SELECT * FROM comments WHERE (EXTRACT(MONTH FROM published_date)) != '04' ORDER BY published_date ASC;

SQL dopyt, ktorý vráti v jednom stĺpci všetky dátumy, kedy bol publikovaný nejaký komentár, alebo ohlásený nejaký bug, ale tak, aby sa neodstránili duplikáty. Je ich 18.
SELECT published_date FROM comments UNION ALL
SELECT reported_date FROM bug;

SQL dopyt, ktorý vráti meno a priezvisko autora najnovšieho komentáru, vo výsledku nech je jeden stĺpec s názvom “MENO” obsahujúci meno a priezvisko oddelené medzerou. Je to Rachel Clark
SELECT (name || ' ' ||surname) AS "MENO"
FROM system_user
INNER JOIN comments “ON system_user.login = comments.login” alebo “USING(login)” WHERE comments.published_date =
(SELECT MAX(published_date) FROM comments);

SQL dopyt, ktory vrati zoznam pouzivatelov/pouzivatela (iba name a surname) s najviac ohlasenymi bugmi. Pouzivatelov s rovnakym menom a priezviskom moze byt viac. Vo vzorovej db je to iba Julia Jackson.
NEVIEM ESTE ?
SELECT name,surname
FROM system_user
where system_user.login = (select reporter_login from(
select reporter_login,COUNT(*)
from bug
group by reporter_login
HAVING COUNT(*) = (select max(pocet)
from (select reporter_login,COUNT(*) as pocet from bug
group by reporter_login))));

Napíšte sql dopyt ktorý vráti pouzivatelov, ktorých meno začína na R alebo na S. Výsledný stĺpec pomenujte „POcet“. Vo vzorovej tabulke DB je takých pouzívatelov 3.
SELECT COUNT(*) AS "POCET" FROM system_user s WHERE s.name LIKE 'R%' OR s.name LIKE 'S%';

Napíšte sql dopyt, ktorý vráti opisy(description) tých bugov, ktoré boli ohlásené v apríli 2016 zoradené podla dátumu ohlásenia(od najstartsieho po najnovsi). Formát dátumu v DV je DD.MM.YYYY. Vo vzorovej tabulke je ich 6 a prvý z nich je opis „Formatovanie
v sprave nie je zachovane.
SELECT description FROM bug b
WHERE EXTRACT(MONTH FROM b.reported_date) = '04' AND EXTRACT(YEAR FROM b.reported_date) = '2016'
ORDER BY b.reported_date ASC;

Napíšte sql dopyt ktorý vráti pouzivatela, ktoreho priezvisko je prvé podla abecedy. Zobrazte v jednom stlpci jeho meno, priezvisko oddelené medzerou, a v druhom stlpci jeho rolu.Vo vzorovej DB je to Jose Campbell.

SELECT (name || ' ' || surname), role FROM system_user where surname = (SELECT MIN(surname) FROM system_user);

Napíšte sql dopyt ktorý vráti počet bugov, ktoré boli zverejnené v apríli ktoréhokolvek roku. Stĺpec s vysledkom nazvite „BUGY“. Vo vzorovej db je ich 6.
SELECT COUNT(*) AS "BUGY" FROM bug WHERE EXTRACT(MONTH FROM bug.reported_date) = '04';

Napíšte sql dopyt ktorý vráti zoznam pouzivatelov (meno,priezvisko) a pocet ich komentarov z roku 2016(nazov stlpca „POCET2016“. Zobrazte aj tych pouzivatelov ktorí v danom roku nemaju ziadny komentar, pricom pri nich ma byt pocet samozrejme0; ale nezobrazte tych kt. maju v danom roku viac ako 2 komentare.(cize zobrazte iba tych, ktori maju 0,1 komentarov.
SELECT name, surname, count(*) as POCET2016 from system_user
full outer join comments c
using(login)
where EXTRACT(YEAR FROM c.published_date) = '2016' group by name,surname
HAVING COUNT(*)=1
union
SELECT name, surname, 0 as POCET2016 from system_user left outer join comments c using(login)
where c.id IS NULL;

Napíšte sql dopyt ktorý vráti pouzivatelov ktorí neohlásili ziaden bug. Zobrazte ich meno, priezvisko v takomto poradí. Vo vzorovej DB je takych pouzivatelov 9, okrem Paul Stewart a jednej Julia Jackson.
SELECT name, surname FROM system_user s
FULL OUTER JOIN bug b ON s.login = b.reporter_login WHERE b.reporter_login is NULL;

Napíšte sql dopyt ktorý vráti zoznam nazvov features(stlpec title), ktoré nemajú ziaden taký bug, na ktorý bol daný komentar obsahujuci v texte nejakú obmenu retazca „fix“ (na velkosti retazca fix nezalezi). Vo vzorovej db su to E-mail notifications, Social networks integration, Performance monitoring, HipChat notifikacie.
SELECT name as title from feature
minus
select name as title from feature
inner join bug_feature
on bug_feature.feature_id=feature.id inner join comments
using(bug_id)
where upper(comments.text) like '%FIX%';