MUNI MED

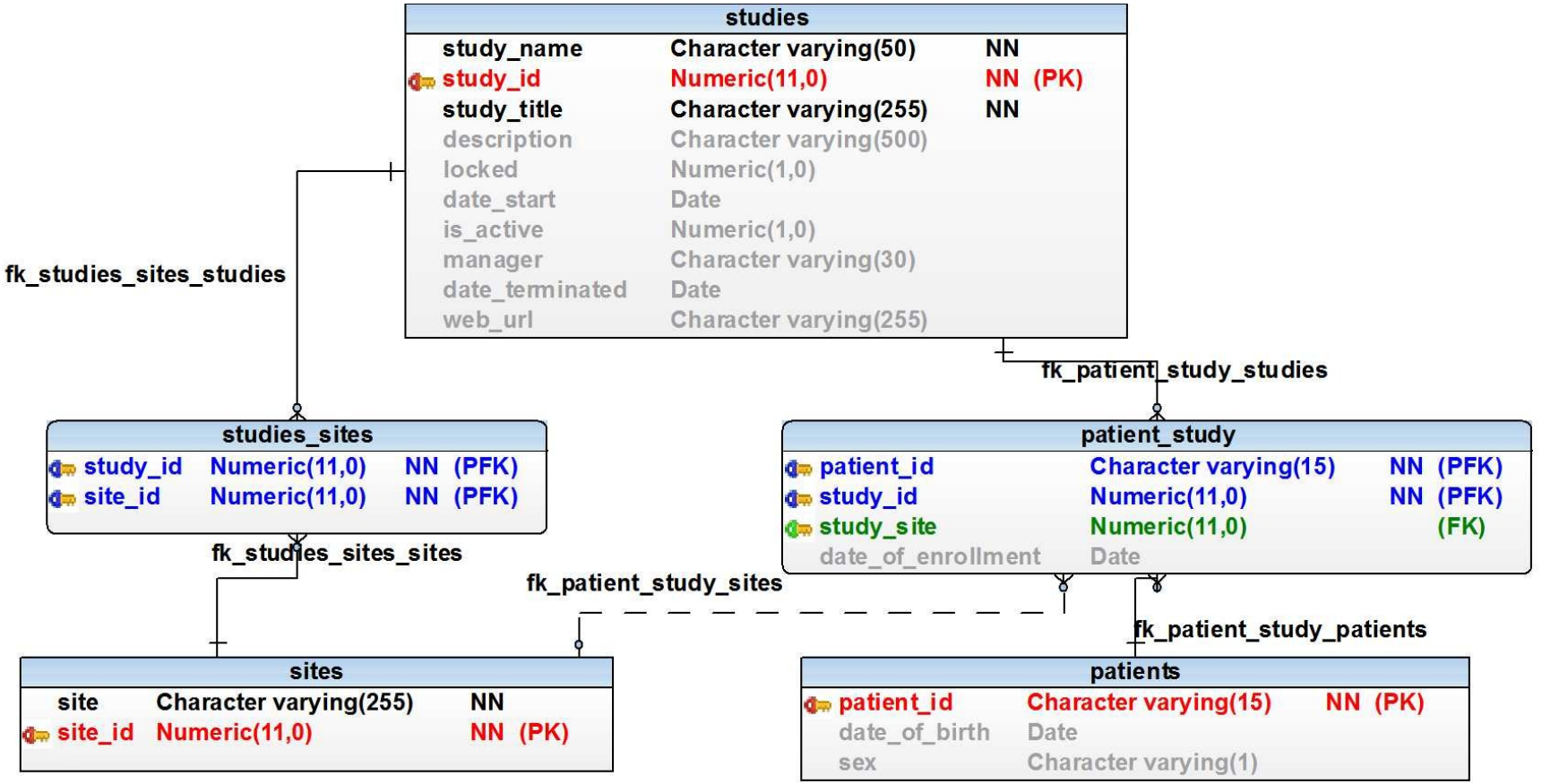
Institut biostatistiky a analýz

Databázové systémy v biomedicíně

Lekce V – Zanořené dotazy

Another data model

patients – studies m-n => „mezitabulka“ PATIENT_STUDY
studies – sites m-n => „mezitabulka“ STUDIES_SITES



Cvičení

Zjistěte počet pacientů v jednotlivých studiích
How many patients are enrolled in each study
Result: STUDY_NAME, number of patients

Zjistěte počet pacientů dle pohlaví v jednotlivých studiích
How many patients are enrolled in each study grouped by sex
Result: STUDY_NAME, sex, number of patients

Zjistěte počet zapojených pracovišť do jednotlivých studií
How many sites participate in each study?
Result: STUDY_NAME, number of sites

Vypište pracoviště zapojená do více studií
Select all sites, which participate in more than 1 study
SITE, počet studií

Vypište všechny studie a počet zařazených pacientů v jednotlivých letech
Select all studies and number of enrolled patients in each year
STUDY_NAME, rok(DATE_OF_ENROLLMENT), počet pacientů

Problém časové řady

```
SELECT s.study_name, to_char(date_of_enrollment, 'yyyy'), count(*)
FROM patient_study ps
JOIN studies s ON ps.study_id  = s.study_id
WHERE study_name  = 'IKARUS'
GROUP BY s.study_id,s.study_name, to_char(date_of_enrollment, 'yyyy')
ORDER BY s.study_name, to_char(date_of_enrollment, 'yyyy')
```

```
SELECT * FROM roky
```

```
SELECT study_name , a FROM studies s
JOIN roky r ON 1=1
WHERE study_name  = 'IKARUS'
ORDER BY s.study_name , r.a
```

```
SELECT study_name , a, count(ps.patient_id) FROM studies s
JOIN roky r ON 1=1
LEFT JOIN patient_study ps ON ps.study_id  = s.study_id
AND to_char(ps.date_of_enrollment, 'yyyy')::integer = r.a
WHERE study_name  = 'IKARUS'
GROUP BY s.study_id,s.study_name, r.a
ORDER BY s.study_name, r.a
```

MUNI MED Institut biostatistiky a analýz

Zanořené dotazy

Nested queries / Subqueries

Poddotazy SQL / subquery

Zanořené dotazy / Subqueries

- uzavřené v kulatých závorkách **() /** *enclosed in brackets*
- poddotazem je myšlen příkaz SELECT

SELECT column
FROM table
WHERE condition

GROUP BY
HAVING
ORDER BY

- místo názvu sloupce
- místo názvu tabulky
- v sekci WHERE

Subdotazy SQL - místo sloupce

Vnořený dotaz na pozici sloupce musí vrátit **právě jeden řádek a právě jeden sloupec!**
This type a subquery must return just one row and one column

```
SELECT COUNT(student_uco),
        (SELECT COUNT (*) FROM student)
FROM vyuka;

SELECT COUNT(patient_id),
        (SELECT COUNT  (*) FROM  patients)
FROM patient_study;
```

Subdotazy SQL - místo sloupce

CVIČENÍ:

Napište dotaz, který vrátí seznam všech studentů,
počet jejich registrovaných předmětů
a kolik je to procent ze všech dostupných předmětů.

Create a query, which return a list of all students with number their registered subjects and compute percent from all subjects (= all rows from table předmet)

Subdotazy SQL - místo sloupce

CVIČENÍ:

Napište dotaz, který vrátí seznam všech studentů,
počet jejich registrovaných předmětů
a kolik je to procent ze všech dostupných předmětů

```
SELECT s.uco,  COUNT(v.predmet_id),
        ROUND(100.0 * (COUNT(v.predmet_id)) /
        (SELECT COUNT(*) FROM predmet) )
FROM student s LEFT JOIN vyuka v
ON s.uco=v.student_uco
GROUP BY s.uco;
```

Zanořený dotaz – místo názvu tabulky

Poddotaz na pozici FROM nahrazuje tabulku.
V postgreSQL musí být poddotaz na pozici tabulky **VŽDY** pojmenován!

Subquery instead of a name of the table must have a name/acronym

```
SELECT  COUNT(*)  FROM (
   SELECT study_id, COUNT(patient_id)
     FROM patient_study GROUP BY study_id
   ) sub
```

Zanořený dotaz – místo názvu tabulky

CVIČENÍ:

Napište dotaz, který vrátí seznam studentů, kteří jsou registrováni do více než jednoho předmětu.

Create a query, which select list of students registered in more than 1 subject

Zanořený dotaz – místo názvu tabulky

CVIČENÍ:

Varianta 1 (variant with subquery)

```
SELECT * FROM (
        SELECT s.firstname, s.lastname, s.uco, COUNT(v.predmet_id) pocet
        FROM student s JOIN vyuka v ON s.uco=v.student_uco
        GROUP BY s.firstname, s.lastname, s.uco) sub
WHERE pocet>1;
```

Varianta 2 (variant with HAVING)

```
SELECT s.firstname, s.lastname, s.uco, COUNT(v.predmet_id)
        FROM student s JOIN vyuka v ON s.uco=v.student_uco
        GROUP BY s.firstname, s.lastname, s.uco
        HAVING COUNT(v.predmet_id)>1
        ORDER BY s.firstname;
```

Vnořený dotaz za WHERE

Varianty:

A)

- WHERE sloupec = (SELECT sloupec FROM…

  zanořený dotaz musí vrátit **právě 1 řádek a 1 sloupec**

  subquery must return just 1 row and 1 column

B)

- WHERE sloupec = **ANY** (SELECT sloupec FROM…
- WHERE sloupec **IN** (SELECT sloupec FROM …
- WHERE sloupec > **ALL** (SELECT sloupec FROM …

  zanořený dotaz musí vrátit **1 sloupec a libovolný počet řádků**

  subquery must return just 1 column and 0 – n rows

Vnořený dotaz za WHERE

Varianty:

C)

- WHERE **EXISTS** (SELECT * FROM….
- WHERE **NOT EXISTS** (SELECT * FROM…

  zanořený dotaz může vracet **libovolný počet řádků i sloupců**

  subquery can return 0 – n rows, columns are irrelevant

Zanořené dotazy se obvykle propojují s nadřazeným dotazem pomocí podmínky v sekci WHERE

Subqueries usually contain a parent-related condition after WHERE

Vnořený dotaz za WHERE sloupec = (SELECT …

Varianty:

A)

- WHERE sloupec = (SELECT sloupec FROM…

Example:

SELECT * FROM patients

WHERE date_of_birth = (SELECT MAX(date_of_birth) FROM patients);

Vnořený dotaz za WHERE sloupec ANY/IN/ALL

Varianty:
B)

- WHERE sloupec = **ANY** (SELECT sloupec FROM…
- WHERE sloupec **IN** (SELECT sloupec FROM …
- WHERE sloupec > **ALL** (SELECT sloupec FROM …

Example:

SELECT * FROM student

WHERE uco = ANY (SELECT student_uco FROM vyuka WHERE predmet_id=10);

SELECT * FROM student

WHERE uco IN (SELECT student_uco FROM vyuka WHERE predmet_id=10);

Vnořený dotaz za WHERE EXISTS/NOT EXISTS

Varianty:

C)

- WHERE **EXISTS** (SELECT * FROM….
- WHERE **NOT EXISTS** (SELECT * FROM…

Example:

SELECT * FROM student s

WHERE EXISTS (SELECT * FROM vyuka v

WHERE predmet_id=10

AND s.uco=v.student_uco);

Vnořený dotaz za WHERE

Nejmladší student/ youngest student:

```
SELECT * FROM student WHERE  birthdate  = (
      SELECT MAX(birthdate) FROM student);

SELECT * FROM student WHERE
              birthdate >= ALL (
      SELECT birthdate FROM student);

SELECT * FROM student tab1 WHERE  NOT EXISTS  (
     SELECT  * FROM student tab2
     WHERE tab2. birthdate > tab1. birthdate);
```

Pozor na NULL hodnoty !
Beware of NULLs in data

Task: Přepište na nejstarší studenty ***/ rewrite queries to oldest students***

Subdotazy SQL - Vnořený dotaz za WHERE

Task:

Vypište seznam studentů, kteří nemají registrovaný žádný předmět.

Select all students, who have no registered subject

Subdotazy SQL - Vnořený dotaz za WHERE

CVIČENÍ:

Vypište seznam studentů, kteří nemají registrovaný žádný předmět.

```
SELECT * FROM student s
WHERE NOT EXISTS (
                    SELECT * FROM vyuka v
                    WHERE s.uco=v.student_uco
                    );
```

Cvičení / Task

Cvičení

Najděte všechny učitele, kteří nevyučují žádný předmět.

```
SELECT * FROM teacher u
WHERE NOT EXISTS (
                SELECT * FROM predmet p
                WHERE u.teacher_uco=p.teacher_uco);
```

Cvičení

Vypište všechny studenty, kteří mají zapsaný předmět Databáze v biomedicíně i Černou magii. (predmet_id 1 a 10)

Varianta 1

```
SELECT * FROM student s
 WHERE EXISTS (SELECT * FROM vyuka v
                    WHERE s.uco=v.student_uco AND predmet_id=1)
 INTERSECT
 SELECT * FROM student s
 WHERE EXISTS (SELECT * FROM vyuka v
                    WHERE s.uco=v.student_uco AND predmet_id=10);
```

Cvičení

Vypište všechny studenty, kteří mají zapsaný předmět Databáze v biomedicíně i Černou magii. (predmet_id 1 a 10)

Varianta 2)

```
SELECT * FROM student s
 WHERE EXISTS (SELECT * FROM vyuka v
                    WHERE s.uco=v.student_uco AND predmet_id=1)
AND EXISTS (SELECT * FROM vyuka v
                    WHERE s.uco=v.student_uco AND predmet_id=10);
```

Cvičení

Vypište všechny studenty, kteří mají zapsaný předmět Databáze v biomedicíně (predmet_id 1), ale nemají zapsanou Černou magii (predmet_id 10).

Varianta 1)

```
SELECT * FROM student s
 WHERE EXISTS (SELECT * FROM vyuka v
                    WHERE s.uco=v.student_uco AND predmet_id=1)
 INTERSECT
 SELECT * FROM student s
 WHERE NOT EXISTS (SELECT * FROM vyuka v
                    WHERE s.uco=v.student_uco AND predmet_id=10);
```

Cvičení

Vypište všechny studenty, kteří mají zapsaný předmět Databáze v biomedicíně (predmet_id 1), ale nemají zapsanou Černou magii (predmet_id 10).

Varianta 2)

```
SELECT * FROM student s
 WHERE EXISTS (SELECT * FROM vyuka v
                    WHERE s.uco=v.student_uco AND predmet_id=1)
 AND NOT EXISTS (SELECT * FROM vyuka v
                    WHERE s.uco=v.student_uco AND predmet_id=10);
```

Cvičení

Vypište všechna pracoviště, která v roce 2010 nezařadila do studie žádného pacienta.

```
SELECT * FROM sites s
WHERE NOT EXISTS (
        SELECT * FROM patient_study ps
        WHERE EXTRACT(YEAR FROM date_of_enrollment)=2010
        AND ps.study_site=s.site_id
        );
```

Cvičení

Vypište všechna pracoviště, která zařadila pacienta naposledy v roce 2010.

```
1) SELECT * FROM sites s
WHERE EXISTS (
        SELECT * FROM patient_study ps
        WHERE EXTRACT(YEAR FROM date_of_enrollment)=2010
        AND ps.study_site=s.site_id
        )
AND NOT EXISTS (
        SELECT * FROM patient_study ps
        WHERE EXTRACT(YEAR FROM date_of_enrollment)>2010
        AND ps.study_site=s.site_id
        );
```

Cvičení

Vypište všechna pracoviště, která zařadila pacienta naposledy v roce 2010.

```
2) SELECT s.site, s.site_id,
        MAX(EXTRACT(YEAR FROM ps.date_of_enrollment)) rok
FROM sites s JOIN patient_study ps
ON s.site_id=ps.study_site
GROUP BY s.site, s.site_id
HAVING MAX(EXTRACT(YEAR FROM ps.date_of_enrollment))=2010;
```

Cvičení

Najděte předměty, kam se přihlásil alespoň jeden student (muž) a vypište celkový počet přihlášených studentů.

```
SELECT predmet_id, COUNT(*) FROM  vyuka v
WHERE EXISTS  (
        SELECT predmet_id FROM student s2 JOIN vyuka v2
        ON s2.uco = v2.student_uco
        WHERE s2.sex = 'muž' AND v.predmet_id=v2.predmet_id
        )
GROUP BY predmet_id;
```

Homework

Zjistěte počet pacientů v jednotlivých studiích po pracovištích a dle pohlaví
STUDY_NAME, SITE, SEX, počet pacientů