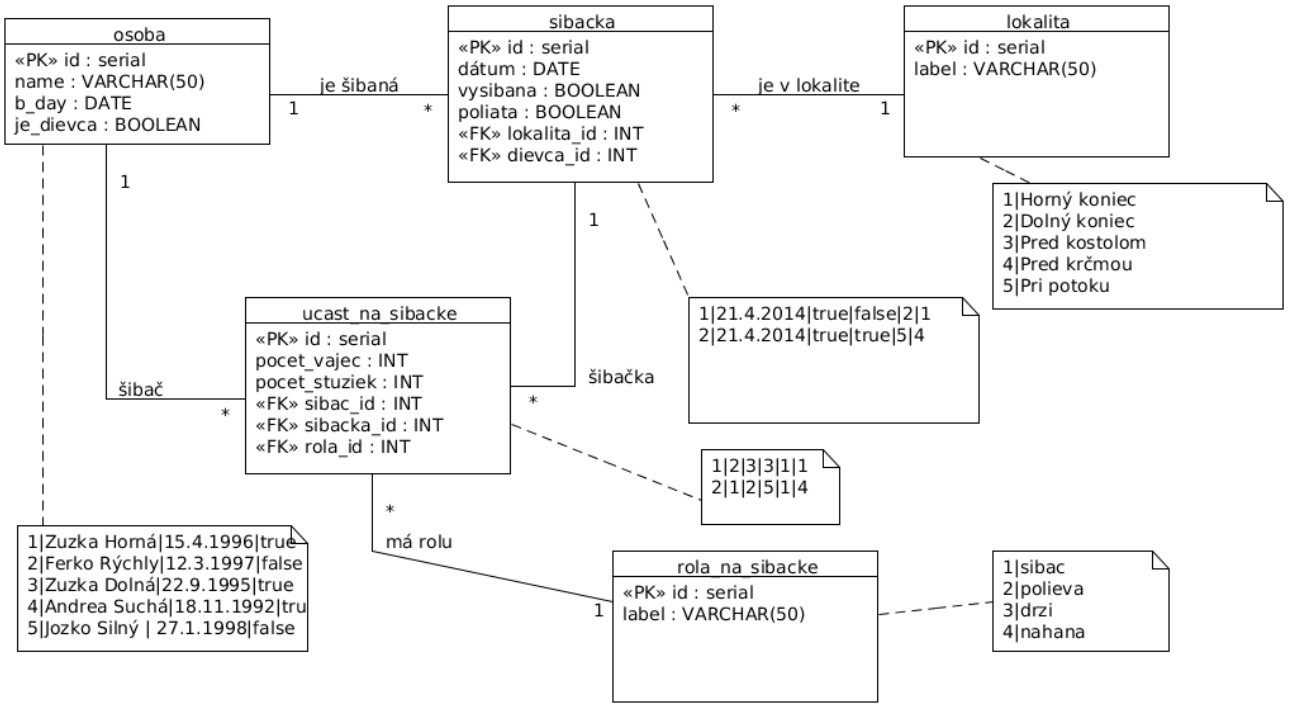
DBS 2015 - riešenia cvičení v šiestom týždni.

Uvažujte nižšie uvedený relačný model ako jedno z možných riešení modelovania šibačky :)



Napíšte SQL dopyty, ktoré poskytujú 4 nasledovné informácie:

1. Priemerný vek všetkých šibačov v rokoch. Šibač je osoba, ktorá sa zúčastnila na nejakej šibačke v roli šibača. HINT: current_date – b_day vráti počet dní medzi dneškom a b_day.

```
SELECT AVG( (current_date – b_day)/365 )  FROM  o
WHERE id IN (
        SELECT sibac_id FROM us WHERE rola_na_sibacke = 1);
```

Toto je otázka, kde kľúčové je to, aby nejakým sposôbom dostali unikátne id šibačov zo šibačiek, teda by nebol žiadny šibac do priemeru zaratány viackrát preto, lebo bol na viacerých šibačkách. Na prednáške som ukazoval toto riešenie, možno to urobil inak.

Nejako by som odlíšil tých, čo zvládli to podeliť tými 365 aby z dní dostali roky, ale nie je to niečo super mega dôležité.

Ak niekto robil JOIN na rolu_v_sibacke a potom porovnáva label, tak OK. Ak robí JOIN a potom porováva id, tak by som to mierne postihoval.

2. Mená dvoch šibačov, ktorí sa zúčastnili na šibačke najväčšieho počtu rôznych dievčat. Vypíšte ich mená a ten počet

```
SELECT meno, COUNT(DISTINCT dievca_id) AS pocet FROM Sibacka s
  JOIN ucast_na_sibacke us ON us.sibacka_id = s.id
  JOIN osoba o ON us.sibac_id = o.id
GROUP BY sibac_id, meno
ORDER BY pocet DESC
LIMIT 2;
```

Toto je otázka zameraná na správne použitie kombinácie GROUP BY + COUNT(DISTINCT) a ORDER BY + LIMIT. V GROUP BY môžu mať osoba.id alebo sibac_id alebo nejaku kombináciu s týmito dvoma atribútmi, ktoré určujú šibača. Tu očakávam všelijaké krkolomné riešenia cez subselecty. Aj keď budú fungovať správne, tak by nemali byť za plný počet bodov.

3. Pre každé dievča vypíšte jej id, meno a celkový počet odmien, ktoré rozdala šibačom. Odmeny sú vajcia a stužky. Dievča je osoba, ktorá má atribút 'je_dievča?' nastavený na true. Výpis nech naozaj obsahuje všetky dievčatá. HINT: COAALESCE(***value*** [, …]), The COALESCE function returns the first of its arguments that is not null. Null is returned only if all arguments are null. It is often used to substitute a default value for null values when data is retrieved for display

```
SELECT o.id, o.name, SUM(COALESCE(pocet_vajec, 0) + COALESCE(pocet_stuziek, 0))
FROM Osoba o
  LEFT JOIN Sibacka s ON s.dievca_id = o.id
  LEFT JOIN ucast_na_sibacke us ON us.sibacka_id = s.id
WHERE o.je_dievca? IS true
GROUP BY o.id
```

Toto je otázka zameraná na LEFT JOIN. Ak ho tam majú použitý aspoň raz, tak fajn. Keďže robíme LEFT JOIN, tak musíme odfiltrovať chlapcov. Ak tam majú obyčajný JOIN, tak je veľmi zle, keďže som zdôraznil, že tam majú byť naozaj všetky dievčatá (teda aj tie, ktoré neboli vyšibané). Bez COALESCE to nezafunguje pre tie dievčatá, ktoré neboli šibané a teda majú v počet_vajec a počet_stužiek NULL. To by som až tak veľmi nepenalizoval. Dôležitý je ten LEFT JOIN.

4. Labely (názvy) všetkých lokalít, kde došlo aspoň trikrát k šibačke, pri ktorej bolo dievča aj poliate.

```
SELECT label FROM Lokalita l
  JOIN sibacka s ON s.lokalita_id = l.id
WHERE poliata IS true
GROUP BY l.id
HAVING COUNT(*) > 2
```

Toto je otázka zameraná na kombináciu GROUP BY + HAVING. Ak to niekto riešil nejako komplikovane inak, sub-selectom, tak by nemal mať úplne plný počet bodov ani ak to má správne.