SESKUPOVÁNÍ A SOUHRNY

- Pomocná tabulka prodeje:

```
CREATE TABLE IF NOT EXISTS prodeje (
id INT UNSIGNED AUTO_INCREMENT PRIMARY KEY,
skupina VARCHAR(255) NOT NULL,
pocet INT NOT NULL,
datum DATE NOT NULL
);

INSERT INTO prodeje (skupina, pocet, datum) VALUES ('Pračky', 7,
'2018-01-05'),('Ledničky', 4, '2018-01,05'),('Ledničky', 2,
'2018-01-21'),('Pračky', 3, '2018-01-25'), ('Pračky', 4, '2018-
02-02'), ('Ledničky', 6, '2018-02-02'), ('Ledničky', 3, '2018-02-
05'), ('Pračky', 3, '2018-02-08');
```

SESKUPOVÁNÍ A SOUHRNY

- Sečtení prodaných kusů pro dané skupiny výrobků:

```
SELECT skupina,SUM(pocet) AS celkem
FROM prodeje
GROUP BY skupina
```

- Seskupování po dvojici hodnot:

```
SELECT skupina,SUM(pocet) AS celkem
FROM prodeje
GROUP BY skupina, datum
```

- Vynechání GROUP BY

```
SELECT SUM(pocet) AS celkem
FROM prodeje
```

SESKUPOVÁNÍ A SOUHRNY

- Sjednocení řádků bez součtu

```
SELECT datum
FROM prodeje
GROUP BY datum
```

- je efektivnější náhrada DISTINCT:

```
SELECT DISTINCT datum
FROM prodeje
```

- Seskupování řádků s hodnotami NULL:
  - konstanta NULL se zde chápe jako jakákoliv jiná hodnota

VÝBĚR SLOUPCŮ V DOTAZU SE SESKUPOVÁNÍM

- V klauzuli GROUP BY by měly být všechny vypisované sloupce, které nepoužívají agregační funkci. Toto je špatně (není jasní, zda jde o pračky, či ledničky):

```
SELECT datum, SUM(pocet) AS celkem
FROM prodeje
GROUP BY datum, skupina
```

- Obrácený přístup v MSSQL vede na chybu, v MySQL zobrazí nesmysl:

```
SELECT skupina, datum, SUM(pocet) AS celkem
FROM prodeje
GROUP BY datum
```

AGREGAČNÍ FUNKCE

- Součet – SUM()
- Průměr – AVG()
- Minimum – MIN()
- Maximum – MAX()
- Počet řádků – COUNT():

```
SELECT COUNT(*) AS celkem_radku FROM prodeje
```

- Hodnota NULL se u COUNT() nepočítá:

```
SELECT COUNT(obec), COUNT(krajske_mesto)
FROM obce
```

AGREGAČNÍ FUNKCE

- Nejmenší a největší hodnoty z tabulky prodeje:

```
SELECT MIN(pocet) AS cislo_min,
MAX(pocet) AS cislo_max, MIN(datum) AS
datum_min, MAX(datum) AS datum_max,
MIN(skupina) as text_min, MAX(skupina)
AS text_max

FROM prodeje
```

AGREGAČNÍ FUNKCE

- rozdíl mezi count(*) a COUNT(sloupec):

```
SELECT count(*), count(id) FROM `obce` GROUP BY krajske_mesto
```

  - Pokud je v sloupci hodnota NULL, COUNT(sloupec) jí nezapočítává, COUNT(*) ano.
- Klauzule DISTINCT v agregační funkci:
  - `SELECT count(*), count(krajske_mesto), count(DISTINCT krajske_mesto) FROM obce`

PŘÍKLADY

- Z tabuky odpracovaných hodin zobrazte součet hodin pro jednotlivé zaměstnance:

```
SELECT os_cislo, SUM(pocet_hodin) AS celkem
FROM hodiny GROUP BY os_cislo
```

- Z tabulky odpracovaných hodin zobrazte pro jednotlivé zaměstnance počet odpracovaných dní a průměrný počet hodin:

```
SELECT os_cislo, COUNT(pocet_hodin) AS
pocet, AVG(pocet_hodin*1.0) AS prumer FROM
hodiny GROUP BY os_cislo
```

PŘÍKLADY

- Zobraz počet zaměstnanců, kteří jsou zapsáni v tabulce odpracovaných hodin:

```
SELECT COUNT(DISTINCT os_cislo) AS pocet
FROM hodiny
```

- Z tabulky odpracovaných hodin zobraz nejmenší a největší počet hodin pro jednotlivé zaměstnance:

```
SELECT os_cislo, MIN(pocet_hodin) AS
min, MAX(pocet_hodin) AS max FROM hodiny
GROUP BY os_cislo
```

ZOBRAZENÍ CELKOVÉHO SOUHRNU

- Seskupení podle jednoho sloupce:

```
SELECT vyrobky.nazev, SUM(polozky.mnozstvi)
FROM polozky INNER JOIN vyrobky ON
vyrobky.cislo=polozky.cislo_vyrobku GROUP BY
vyrobky.nazev WITH ROLLUP
```

- Vylepšení o zobrazení slova celkem namísto NULL:

```
SELECT IFNULL(vyrobky.nazev, 'celkem'),
SUM(polozky.mnozstvi) FROM polozky INNER JOIN
vyrobky ON
vyrobky.cislo=polozky.cislo_vyrobku GROUP BY
vyrobky.nazev WITH ROLLUP
```

ŘAZENÍ

- Při seskupování dle více sloupců se řazení provede zleva doprava:

```
SELECT skupina, datum FROM prodeje
GROUP BY skupina, datum
```

- Daný sloupec lze upřednostnit pomocí ORDER BY na konci dotazu:

```
SELECT skupina, datum FROM prodeje
GROUP BY skupina, datum ORDER BY datum
```

PŘEDBĚŽNÁ FILTRACE

- Klauzule WHERE v dotaze se souhrnem znamená, že se filtrace provede ještě dříve, než se začnou řádky seskupovat.
- Součet prodaných kusů a omezení výběru na leden 2018:

```
SELECT skupina, SUM(pocet) AS celkem FROM prodeje WHERE datum BETWEEN '2018-01-01' AND '2018-01-31' GROUP BY skupina
```

- Součet řádků s hodnotou NULL:

```
SELECT COUNT(*) FROM obce WHERE krajske_mesto IS NULL;
```

NÁSLEDNÁ FILTRACE

- Klauzuli HAVING používáme pro možnost výběru řádků až po seskupení.
- Z tabulky prodejů provedeme sumaci po datumech, ale ve výsledku zobrazíme jen dny, kde se prodalo 10 a více ks:

```
SELECT datum, SUM(pocet) AS celkem
FROM prodeje GROUP BY datum HAVING
SUM(pocet) >=10;
```

WHERE VS HAVING

Pomocná tabulka drubez:

```
CREATE TABLE IF NOT EXISTS drubez (
id INT UNSIGNED AUTO_INCREMENT PRIMARY KEY,
druh VARCHAR(255) NOT NULL,
pocet INT NOT NULL
);
```

- `INSERT INTO drubez (druh, pocet) VALUES ('slepice', 15), ('slepice', 12), ('kachny', 14), ('kachny', 8), ('husy', 7), ('husy', 6), ('krůty', 4), ('krůty', 3);`

WHERE VS HAVING

- Předběžná filtrace (WHERE):

```
SELECT druh, SUM(pocet)  AS celkem
FROM drubez WHERE pocet > 10 GROUP
BY druh;
```

- Následná filtrace (HAVING):

```
SELECT druh, SUM(POCET) AS celkem
FROM drubez GROUP BY druh HAVING
SUM(pocet) > 10;
```

KOMBINACE FILTRŮ

- Pořadí:
  1)Předběžná filtrace (WHERE),
  2)seskupení (GROUP BY),
  3)následná filtrace (HAVING).
- Dotaz sečte prodané kusy pro obě skupiny výrobků, omezí výběr na leden 2018 a ve výsledku zobrazí jen řádky, u kterých je celkový součet 10 a více:

```
SELECT skupina, SUM(POCET) AS celkem FROM
prodeje WHERE datum BETWEEN '2018-01-01' AND
'2018-01-31' GROUP BY skupina HAVING SUM(pocet)
>= 10;
```

VÝPOČTY

- Pomocná tabulka výroba:

```
CREATE TABLE IF NOT EXISTS vyroba (
id INT UNSIGNED AUTO_INCREMENT PRIMARY KEY,
kod CHAR(5),
datum DATE NOT NULL,
kusu INT UNSIGNED NOT NULL,
cena DECIMAL(12,2) NOT NULL,
cas INT UNSIGNED NOT NULL
);
INSERT INTO vyroba (kod,datum, kusu, cena, cas) VALUES
('AS787','2018-02-15', 15400, 2.3, 530), ('AS745','2018-02-15',
14821, 3.7, 524), ('AY722','2018-03-02', 14008, 3.7, 512),
('AY400','2018-03-02', 15125, 2.3, 529), ('AU333','2018-03-05',
14872, 2.3, 501), ('AG191','2018-03-05', 15002, 3.7, 517);
```

VÝPOČET PŘED AGREGACÍ

- Chceme znát cenu vyrobených součástek za každý den.
- Během jednoho dne se však na lince vyrábí různé typy souč. s rozdílnou cenou.
- Nejprve spočítáme cenu, poté sečteme:

```
SELECT datum, SUM(kusu*cena) AS
celkem FROM vyroba GROUP BY datum;
```

VÝPOČET PO AGREGACI

- Potřebujeme zjistit průmernou dobu potřebnou k výrobě jedné součástky.
- Pro jednotlivé dny zjistíme celkový počet vyrobených součástek a celkovou provozní dobu.
- Obě hodnoty nakonec vydělíme a vynásobíme 60 pro obdržení počtu sekund potřebných pro výrobu jedné součástky:

```
SELECT datum, SUM(cas)/SUM(kusu) * 60 AS
podil FROM vyroba GROUP BY datum;
```

VZOREC V GROUP BY

- Sečtení vyrobených kusů po měsících:

```
SELECT MONTH(datum)  AS mesic, SUM(kusu) AS celkem
FROM vyroba GROUP BY MONTH(datum);
```

- Sečtení počtu vyrobených součástek podle skupin (skupina = první dva znaky v kódu souč.):

```
SELECT LEFT(kod, 2) AS skupina, SUM(kusu) AS celkem
FROM vyroba GROUP BY LEFT(kod, 2);
```

- Zjištění počtu obratů do 1000Kč a nad:

```
SELECT CASE WHEN obrat<1000 THEN '<1000' ELSE
'>1000' END AS vyse_obratu, COUNT(obrat) AS pocet
FROM obraty GROUP BY  CASE WHEN obrat<1000 THEN
'<1000' ELSE '>1000' END;
```

SOUHRNY Z VÍCE TABULEK

- Výpočet celkových dodaných množství pro jednotlivé výroby:

```
SELECT vyrobky.nazev, SUM(dodavky.mnozstvi)
AS soucet FROM vyrobky INNER JOIN dodavky ON
dodavky.cislo_vyrobku=vyrobky.cislo GROUP BY
vyrobky.nazev;
```

- Dotaz, zjišťující, kolikrát byly dané výrobky prodány:

```
SELECT vyrobky.nazev, COUNT(dodavky.mnozstvi)
AS pocet1, COUNT(*) AS pocet2  FROM vyrobky
INNER JOIN dodavky ON
dodavky.cislo_vyrobku=vyrobky.cislo GROUP BY
vyrobky.nazev;
```

SOUHRNY Z VÍCE TABULEK

- Zjištění data poslední dodávky:

```
SELECT vyrobky.nazev, MAX(dodavky.datum) AS
posledni FROM vyrobky INNER JOIN dodavky ON
dodavky.cislo_vyrobku = vyrobky.cislo GROUP
BY vyrobky.nazev;
```

- Zobrazení úhrnných tržeb:

```
SELECT vyrobky.nazev,
SUM(vyrobky.cena*dodavky.mnozstvi) AS trzba
FROM vyrobky INNER JOIN dodavky ON
dodavky.cislo_vyrobku = vyrobky.cislo GROUP
BY vyrobky.nazev;
```

SOUHRNY Z VÍCE TABULEK

- Úhrnná dodaná množství za rok 2017:

```
SELECT vyrobky.nazev, SUM(dodavky.mnozstvi) AS
celkem FROM vyrobky INNER JOIN dodavky ON
dodavky.cislo_vyrobku = vyrobky.cislo WHERE
dodavky.datum BETWEEN '2017-01-01' AND '2017-12-31'
GROUP BY vyrobky.nazev HAVING
SUM(dodavky.mnozstvi)>200;
```

- Pět výrobků s nejvyššími tržbami:

```
SELECT vyrobky.nazev, ROUND(SUM(vyrobky.cena *
dodavky.mnozstvi),0) AS trzba FROM vyrobky INNER
JOIN dodavky ON dodavky.cislo_vyrobku =
vyrobky.cislo  GROUP BY vyrobky.nazev ORDER BY trzba
DESC LIMIT 5;
```

SOUHRNY Z VÍCE TABULEK

- Zobraz tržby včetně DPH pro jednotlivé zákazníky:

```
SELECT zakaznici.jmeno, SUM(vyrobky.cena + vyrobky.cena
* vyrobky.sazba_dph/100 * polozky.mnozstvi) AS trzba
FROM zakaznici

INNER JOIN faktury ON faktury.odberatel = zakaznici.id

INNER JOIN polozky ON polozky.faktura = faktury.cislo

INNER JOIN vyrobky on vyrobky.cislo =
polozky.cislo_vyrobku

GROUP BY zakaznici.jmeno
```

VNĚJŠÍ SPOJENÍ TABULEK

- Může se objevit hodnota NULL, která řešení dotazu zpravidla zkomplikuje.

```
CREATE TABLE IF NOT EXISTS ucitele (
id INT UNSIGNED AUTO_INCREMENT PRIMARY KEY,
jmeno VARCHAR(255) NOT NULL
);
INSERT INTO ucitele (jmeno) VALUES ('Jaroslav'), ('Miroslav'),
('Ladislav');
CREATE TABLE IF NOT EXISTS predmety (
id INT UNSIGNED AUTO_INCREMENT PRIMARY KEY,
predmet VARCHAR(255) NOT NULL,
ucitel INT NOT NULL,
hodin INT NOT NULL
);
INSERT INTO predmety (ucitel, predmet, hodin) VALUES (1,'Základy jazyka
SQL', 60), (1,'Administrace databází', 75), (2,'Testování software', 60);
```

VNĚJŠÍ SPOJENÍ TABULEK

- Dotaz, počítající pro všechny učitele celkové úvazky:
- SELECT ucitele.jmeno, SUM(predmety.hodin) AS celkem FROM ucitele LEFT OUTER JOIN predmety ON ucitele.id = predmety.ucitel GROUP BY ucitele.jmeno;
- Vylepšete, aby dotazmísto NULL ukazoval u Ladislava 0.
- Zobrazte pomocí COUNT(), kolik předmětů jaký učitel vyučuje.

SPOJENÍ DOTAZŮ SE SOUHRNEM

- Představme si tabulku prodejů, obsahující číslo měsíce, jméno prodejce, výši tržby a další údaje.
- Spočítejme měsíční průměry prodejců zvlášť pro muže a zvlášť pro ženy, kdy tabulka neobsahuje informaci o pohlaví prodejce.

```
SELECT mesic, AVG(trzba) AS prumer , 'muzi' AS typ
FROM data WHERE  zastupce in ('Jaroslav', 'Petr',
'Marek') GROUp BY mesic

UNION

SELECT mesic, AVG(trzba) AS prumer , 'zeny' AS typ
FROM data WHERE  zastupce in ('Jitka', 'Pavla',
'Marie') GROUp BY mesic
```

1)Z tabulky obratů zjistěte počet řádků, součet a průměr obratů a nejvyšší a nejmenší obrat.

2)Z tabulky faktur zjistěte počet zaplacených faktur (vyplněno datum zaplacení).

3)Z tabulky obratů vypočtěte součet obratů a počet řádků pro jednotlivá čísla protiúčtů (sloupec „ucet“)

4)Z tabulky obratů zjistěte počet jednotlivých protiúčtů.

5)Z tabulky obratů vypočítejte průměrný obrat pro jednotlivá čísla protiúčtů a celkový průměr.

1) Z tabulky položek vypočítejte součty částek pro jednotlivé faktury a čísla výrobků, celkové součty částek pro každou fakturu a součet všech částek v celé tabulce.

2) Z tabulky faktur vypočítejte součet částek podle čísla zákazníka. Výpočet bude omezen jen na zaplacené faktury.

3) Z tabulky faktur vypočítejte součet částek podle čísla zákazníka. Dotaz zobrazí pouze řádky se součtem nad 40 000Kč.

4) Z tabulky faktur vypočítejte součet částek podle čísla zákazníka za rok 2018. Dotaz zobrazí pouze řádky se součtem nad 20 000Kč.

5) Z tabulky položek spočítejte pro jednotlivé faktury součet podílů částka/množství.

1)Z tabulky obraty4 po jednotlivých protiúčtech (sloupec účet) zjistěte podíl součtu poplatků a součet obratů v procentech.

2)Z tabulky faktur určete počet neproplacených faktur.

3)Z tabulky obratů vypočtěte součet obratů pro jednotlivé banky.

4)Z tabulky obratů vypočtěte součet obratů po rocích a měsících.

5)Z tabulky obratů vypočtěte součet a počet obratů do 10 000Kč a nad tuto částku.

1) Zobrazte názvy skupin, počet výrobků v každé skupině a průměrnou cenu výrobků ve skupině.
2) Pro jednotlivá jména zákazníků zobrazte celkové nafakturované částky.
3) Pro jednotlivá jména zákazníků zobrazte datum poslední fakturace.
4) Pro jednotlivé názvy výrobků zobrazte počet jejich prodejů, které jsou vyšší, než 200 jednotek.
5) Zobrazte názvy výrobků, pro které je celková fakturovaná částka v tabulce položek vyšší, než 10 000 Kč.
6) Zobrazte názvy skupin a rozdíl mezi nejvyšší a nejnižší cenou u výrobků ve skupině.

- Zobrazte jména jednotlivých zákazníků, názvy výrobků a uhrnná prodaná množství pro jednotlivé zákazníky a výrobky.
- Určete celkový počet předpisů, pro které neexistuje příslušná platba.
- Zobrazte jména všech výrobků a jejich úhrnná prodaná množství.
- Zobrazte jména všech výrobků a počet jejich prodejů.
- Zobrazte příjmení všech zaměstnanců a počet zákazníků, které mají jednotliví zaměstnanci na starosti.
- Pro jednotlivé zákazníky zobrazte celkové nafakturované částky, zvlášť pro neproplacené a zvlášť pro proplacené faktury.