-- 1.Покажите среднюю зарплату сотрудников за каждый год (средняя заработная плата среди тех, кто работал в отчетный период -статистика с начала до 2005 года).

SELECT avg(salary), year(from\_date)

FROM employees.salaries

WHERE YEAR(from\_date) BETWEEN YEAR(from\_date) AND YEAR(to\_date)

AND YEAR(from\_date)<2005

GROUP BY YEAR(from\_date)

ORDER BY YEAR(from\_date)

;

-- 2.Покажите среднюю зарплату сотрудников по каждому отделу. Примечание: принять в расчет только текущие отделы и текущую заработную плату.

SELECT

ede.dept\_no, AVG(salary)

FROM

employees.salaries AS es

INNER JOIN

employees.dept\_emp AS ede USING (emp\_no)

WHERE

CURDATE() BETWEEN es.from\_date AND es.to\_date

AND CURDATE() BETWEEN ede.from\_date AND ede.to\_date

GROUP BY ede.dept\_no

ORDER BY ede.dept\_no

;

-- 3.Покажите среднюю зарплату сотрудников по каждому отделу за каждый год. Примечание: для средней зарплаты отдела Xв году Yнам нужно взять среднее значение всех зарплат в году Yсотрудников,которые были в отделе Xв году Y.

SELECT avg(es.salary), year(es.from\_date), dept\_no

FROM employees.salaries as es

INNER JOIN employees.dept\_emp as ede ON (es.emp\_no=ede.emp\_no AND YEAR(es.from\_date) BETWEEN YEAR(ede.from\_date) AND YEAR(ede.to\_date)

GROUP BY YEAR(es.from\_date), dept\_no

ORDER BY YEAR(es.from\_date)

;

-- 4.Покажите для каждого года самый крупный отдел (по количеству сотрудников) в этом году и его среднюю зарплату

SELECT efd, MAX(count\_em), DEPT\_NO, avgsal

FROM (SELECT COUNT(ede.emp\_no) as count\_em, Year(ES.FROM\_DATE) as efd, dept\_no, avg(salary) as avgsal

FROM employees.dept\_emp as ede

INNER JOIN employees.salaries AS es ON (ede.emp\_no=es.emp\_no AND

YEAR(es.from\_date) BETWEEN YEAR(ede.from\_date) AND YEAR(ede.to\_date) )

GROUP BY YEAR(es.from\_date), dept\_no) as tt1

group by efd

;

-- 5.Покажите подробную информацию о менеджере, который дольше всех исполняет свои обязанности на данный момент.

SELECT ee.emp\_no, CONCAT(ee.first\_name, ' ', ee.last\_name) AS Fulname,

et.title, es.salary, ed.dept\_no, ed.dept\_name, ee.hire\_date, ee.birth\_date,

ee.gender, tl.lang\_name, edm.from\_date AS 'Working Manager from'

FROM employees.dept\_manager as edm

INNER JOIN employees.salaries as es ON (es.emp\_no=edm.emp\_no)

INNER JOIN employees.employees as ee ON (ee.emp\_no=edm.emp\_no)

INNER JOIN employees.departments as ed ON (ed.dept\_no=edm.dept\_no)

INNER JOIN employees.titles as et ON (et.emp\_no=edm.emp\_no)

INNER JOIN tempdb.language as tl ON (tl.lang\_no=ee.lang\_no)

WHERE CURDATE() BETWEEN edm.from\_date AND edm.to\_date

AND CURDATE() BETWEEN es.from\_date AND es.to\_date

AND CURDATE() BETWEEN et.from\_date AND et.to\_date

ORDER BY edm.from\_date ASC

LIMIT 1;

-- 6.Покажите топ-10 нынешних сотрудников компании с наибольшей разницей между их зарплатой и текущей средней зарплатой в их отделе

SELECT emp\_no,salary, ROUND(salary-(SELECT avg(salary)

FROM employees.salaries

WHERE CURDATE() BETWEEN from\_date AND to\_date)) as dif

FROM employees.salaries as es

WHERE CURDATE() BETWEEN from\_date AND to\_date

ORDER BY dif DESC

LIMIT 10

;

-- 7.Из-за кризиса на одно подразделение на своевременную выплату зарплаты выделяется всего 500 тысяч долларов. Правление решило, что низкооплачиваемые сотрудники будут первыми получать зарплату.

-- Показать список всех сотрудников, которые будут вовремя получать зарплату (обратите внимание, что мы должны платить зарплату за один месяц, но в базе данных мы храним годовые суммы).

SELECT \*

FROM (

SELECT emp\_no, dept\_no, salary, SUM(salary/12) OVER (PARTITION BY dept\_no ORDER BY salary, es.from\_date RANGE BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW) AS cum\_sum

FROM employees.salaries as es

INNER JOIN employees.dept\_emp as ede USING (emp\_no)

WHERE CURDATE() BETWEEN es.from\_date AND es.to\_date

AND CURDATE() between ede.from\_date AND ede.to\_date

ORDER BY dept\_no,salary) as tab

WHERE cum\_sum<500000;

CREATE DATABASE school;

USE school;

CREATE TABLE students (

student\_no INT NOT NULL ,

teacher\_no INT,

course\_no INT,

student\_name VARCHAR(50),

email VARCHAR(50),

birth\_date DATE NOT NULL,

PRIMARY KEY(student\_no, birth\_date)

)

PARTITION BY RANGE (YEAR(birth\_date)) (

PARTITION p\_less VALUES LESS THAN (1983),

PARTITION p1983 VALUES LESS THAN (1984),

PARTITION p1984 VALUES LESS THAN (1985),

PARTITION p1985 VALUES LESS THAN (1986),

PARTITION p1986 VALUES LESS THAN (1987),

PARTITION p1987 VALUES LESS THAN (1988),

PARTITION p1988 VALUES LESS THAN (1989),

PARTITION p1989 VALUES LESS THAN (1990),

PARTITION p1990 VALUES LESS THAN (1991),

PARTITION p1991 VALUES LESS THAN (1992),

PARTITION p1992 VALUES LESS THAN (1993),

PARTITION p1993 VALUES LESS THAN (1994),

PARTITION p1994 VALUES LESS THAN (1995),

PARTITION p1995 VALUES LESS THAN (1996),

PARTITION p1996 VALUES LESS THAN (1997),

PARTITION p1997 VALUES LESS THAN (1998),

PARTITION p1998 VALUES LESS THAN (1999),

PARTITION p1999 VALUES LESS THAN (2000),

PARTITION p2000 VALUES LESS THAN (2001),

PARTITION p2001 VALUES LESS THAN (2002),

PARTITION p2002 VALUES LESS THAN (2003),

PARTITION p2003 VALUES LESS THAN (2004),

PARTITION p2004 VALUES LESS THAN (2005),

PARTITION p2005 VALUES LESS THAN (2006),

PARTITION p2006 VALUES LESS THAN (2007),

PARTITION p2007 VALUES LESS THAN (2008)

);

CREATE INDEX idx\_email ON students(email);

CREATE TABLE teachers (

teacher\_no INT,

teacher\_name VARCHAR(50),

phone\_no VARCHAR(13)

);

CREATE UNIQUE INDEX idx\_phone\_no ON teachers(phone\_no);

CREATE TABLE courses (

course\_no INT,

course\_name VARCHAR(100),

start\_date DATE,

end\_date DATE

);

2.;

INSERT INTO students (student\_no, teacher\_no, course\_no, student\_name, email, birth\_date)

VALUES

('10001', '7001', '101', 'Adam Arhold', 'adam\_arh1999@gmail.com', '1999-01-21'),

('10002', '7005', '103', 'Viktoria Viktorova', 'vivi1995@gmail.com', '1995-09-09'),

('10003', '7005', '104', 'Valentina Studak', 'valstu2107@gmail.com', '1998-07-21'),

('10004', '7002', '102', 'Bogdan Lesley', 'lesley\_b@ukr.net', '2000-12-08'),

('10005', '7002', '102', 'Anatoliy Sid', 'anatol\_rig1989', '1989-02-28'),

('10006', '7003', '103', 'Petro Pavlik', 'petpavpav@gmail.com', '1998-07-13'),

('10007', '7004', '114', 'Polina Romanuk', 'polrom\_rovno@meta.ua', '2002-09-07'),

('10008', '7005', '104', 'Galina Pozniak', 'galinkaggg@gmail.com', '2004-03-21'),

('10009', '7004', '114', 'Natalia Makarenko', 'makaren\_nat@gmail.com', '2002-05-31'),

('10010', '7001', '101', 'Evgeniy Polezhaiko', 'evgen\_policmen1999@gmail.com', '1999-01-31');

SELECT \*

FROM

students;

INSERT INTO teachers

VALUES

('7001', 'Igor Pavchenko', '+380931234567'),

('7002', 'Pavel Gonchar', '+380501234567'),

('7003', 'Inna Petrenko', '+380731234567'),

('7004', 'Andriy Omelchenko', '+380916019444'),

('7005', 'Dmitro Rozhin', '+380991234567'),

('7006', 'Darina Marchuk', '+380997654321'),

('7101', 'Pavel Petrunek', '+380954359632'),

('7102', 'Olga Andrienko', '+380971268834'),

('7103', 'Anna Korniuk', '+380991881271');

SELECT \*

FROM

teachers;

INSERT INTO courses

VALUES

('101', 'PHP Development', '2020-09-01', '2022-05-31'),

('102', 'Data Analyst', '2021-03-15', '2021-09-21'),

('103', 'QA', '2020-09-01', '2023-02-25'),

('104', 'Business Inteligence', '2021-10-16', '2022-09-01'),

('105', 'Вебпрограмування з Python та JavaScript CS50', '2022-03-15', '2022-09-15'),

('106', 'Основи Web UI розробки 2022', '2022-09-15', '2023-05-01'),

('114', "Англійська для кар'єрного зростання", '2022-05-16', '2023-01-30');

SELECT \*

FROM courses;

3. ;

SELECT \*

FROM school.students

WHERE birth\_date BETWEEN '2002-01-01' AND '2002-12-31'

;

EXPLAIN SELECT \*

FROM school.students

WHERE birth\_date BETWEEN '2002-01-01' AND '2002-12-31'

;

/\*# id, select\_type, table, partitions, type, possible\_keys, key, key\_len, ref, rows, filtered, Extra

'1', 'SIMPLE', 'students', 'p2002', 'ALL', NULL, NULL, NULL, NULL, '2', '50.00', 'Using where' \*/

4.;

EXPLAIN SELECT \*

FROM teachers

WHERE phone\_no = '+380916019444';

/\*id, select\_type, table, partitions, type, possible\_keys, key, key\_len, ref, rows, filtered, Extra

' 1', 'SIMPLE', 'teachers', NULL, 'const', 'idx\_phone\_no', 'idx\_phone\_no', '55', 'const', '1', '100.00', NULL \*/

ALTER TABLE school.teachers

ALTER INDEX idx\_phone\_no INVISIBLE;

EXPLAIN SELECT \*

FROM teachers

WHERE phone\_no = '+380916019444';

# id, select\_type, table, partitions, type, possible\_keys, key, key\_len, ref, rows, filtered, Extra

# '1', 'SIMPLE', 'teachers', NULL, ALL', NULL, NULL, NULL, NULL, '9', '11.11', 'Using where'

ALTER TABLE school.teachers

ALTER INDEX idx\_phone\_no VISIBLE

5.;

INSERT INTO students (student\_no, teacher\_no, course\_no, student\_name, email, birth\_date)

VALUES

('10013', '7001', '101', 'Adam Arhold', 'adam\_arh1999@gmail.com', '1999-01-21'),

('10011', '7001', '101', 'Adam Arhold', 'adam\_arh1999@gmail.com', '1999-01-21'),

('10012', '7001', '101', 'Adam Arhold', 'adam\_arh1999@gmail.com', '1999-01-21');

SElECT \*

FROM students;

6;

SELECT DISTINCT ss.\*

FROM students as ss

INNER JOIN students as ss1 ON (

ss.student\_no<>ss1.student\_no AND

ss.teacher\_no=ss1.teacher\_no AND

ss.course\_no=ss1.course\_no AND

ss.student\_name=ss1.student\_name AND

ss.email=ss1.email AND

ss.birth\_date=ss1.birth\_date);