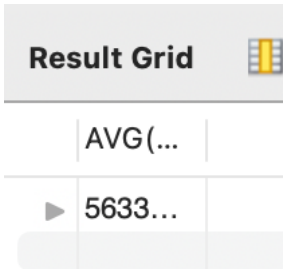
Last Homework Solutions

Problem 1:

```
INSERT INTO
locations(location_id,street_address,postal_code,city,state_province,c
ountry_id)
VALUES (3000, '100 Avda Leonardo Da Vinci',NULL,'Punta del
Este','Maldonado','UR');

INSERT INTO departments(department_id, department_name,location_id)
VALUES (12, 'AppDev',3000);

INSERT INTO jobs(job_id,job_title,min_salary,max_salary)
VALUES (20,'CIO',5000.00,10000.00);

INSERT INTO
employees(employee_id,first_name,last_name,email,phone_number,hire_dat
e,job_id,salary,manager_id,department_id)
VALUES
(207,'Leonardo','Lima','Leonardo.Lima@mu.com',NULL,'2024-06-01',20,750
0.00,100,12);

INSERT INTO
employees(employee_id,first_name,last_name,email,phone_number,
hire_date,job_id,salary,manager_id,department_id)
VALUES
(208,'Julio','Tejera','Julio.Tejeraa@mu.com',NULL,'2024-06-01',9,5000.
00,207,12);

INSERT INTO
employees(employee_id,first_name,last_name,email,phone_number,hire_
date,job_id,salary,manager_id,department_id)
VALUES
(209,'Marcos','Robello','Marcos.Robello@mu.com',NULL,'2024-06-01',3,40
00.00,207,12);
```

Problem 2:

```
SELECT AVG(salary) AS average
FROM employees JOIN jobs USING(job_id)
WHERE job_title = 'programmer'
GROUP BY job_title;
```



Problem 3:

```
SELECT job_title AS 'Title', MAX(salary) AS 'maximum salary'
FROM employees JOIN jobs USING(job_id)
GROUP BY job_title;
```

Result Grid | Filter Rows: Search | Exp

Title	maximum salary
President	24000.00
Administration Vice President	17000.00
Accountant	9000.00
Finance Manager	12000.00
Human Resources Representative	6500.00
Programmer	9000.00
Marketing Manager	13000.00
Marketing Representative	6000.00
Public Relations Representative	10000.00
Purchasing Clerk	3100.00
Purchasing Manager	11000.00
Sales Manager	14000.00
Sales Representative	8600.00
Shipping Clerk	4000.00

Problem 4:

```
SELECT e.first_name, e.last_name, count
FROM employees e JOIN
      (SELECT employee_id, COUNT(*) AS count FROM
      dependents JOIN employees USING(employee_id)
      GROUP BY employee_id) AS ct
    USING (employee_id)
ORDER BY count  DESC
LIMIT 1;
```

first_name	last_name	count
Leonardo	Lima	3

Problem 5:

```
SELECT
      e.department_id,
      department_name,
      MAX(salary)
FROM
      employees e INNER JOIN departments d USING( department_id)
GROUP BY
      e.department_id
HAVING
      MAX(salary) <= 8000
ORDER BY
      MAX(salary);
```

department...	department_name	MAX(salary)
1	Administration	4400.00
4	Human Resources	6500.00
12	AppDev	7500.00

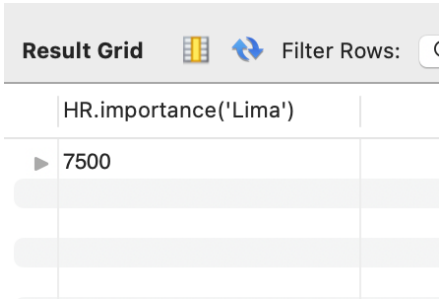
Problem 6:

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `average
salary`(my_department_name VARCHAR(30) )
BEGIN
SELECT AVG(e.salary)
FROM employees e JOIN departments d USING (department_id)
WHERE d.department_name = my_department_name
```

```
GROUP BY department_id;
END
```

Problem 7:

```
CREATE DEFINER=`root`@`localhost` FUNCTION `importance`(manager
VARCHAR(30)) RETURNS int
    READS SQL DATA
BEGIN
DECLARE RETVAL INTEGER;
SELECT SUM(e.salary)
INTO RETVAL
FROM employees e JOIN employees manager ON e.manager_id =
manager.employee_id
WHERE e.last_name = manager
GROUP BY e.employee_id;
RETURN RETVAL;
END
```



Problem 8:

$\{A\}^+ = \{A\}$
$\{B\}^+ = \{B\}$
$\{C\}^+ = \{C\}$
$\{D\}^+ = \{D\}$
$\{E\}^+ = \{E\}$
$\{F\}^+ = \{F, A, B, C, E\}$

$\{A, B\}^+ = \{A, B, C, E\}$
$\{A, C\}^+ = \{A, C\}$
$\{A, D\}^+ = \{A, D\}$
$\{A, E\}^+ = \{A, E\}$
$\{A, F\}^+ = \{A, F, B, C, E\}$
$\{B, C\}^+ = \{B, C\}$
$\{B, D\}^+ = \{B, D\}$
$\{B, E\}^+ = \{B, E\}$
$\{B, F\}^+ = \{B, F, A, E\}$
$\{C, D\}^+ = \{C, D, E\}$
$\{C, E\}^+ = \{C, E\}$
$\{C, F\}^+ = \{C, F, A, B, E\}$

$\{D, E\}^+ = \{D, E\}$
$\{D, F\}^+ = \{D, F, A, B, C, E\}$ key
$\{E, F\}^+ = \{E, F, A, B, C\}$.

Problem 9:

To be in Boyce-Codd normal form, each FD needs to have a super-key on the right side. Since $A, B \nrightarrow E$, $\{A, B\}$ is definitely not a super-key and the first FD $A, B \to C$ violates the Boyce-Codd condition. We can add to the FD and obtain a new FD $A, B \to C, D$, which combines the first and second original FD. We then apply the decomposition algorithm (Algorithm 3.20) . Since

$$\{A, B\}^+ = \{A, B, C, D\},$$

we create a new table

$$R_1(A, B, C, D)$$

with projected FDs $A, B \to C, D$, showing that A, B is a key. This table is in BCNF.

The second relation consists of the left side of the violating FC and the other attributes, i.e.

$$R_2(A, B, E).$$

This relation has no FDs! In fact, the third original FD $E \to D$ is no longer reconstructable from the decomposition. This is nothing unusual, though of course regrettable. As R_2 does not have any more FDs, it is by default in BCNF.

Problem 10:

Since the writes to page x are done by transactions 2, 3, and 1 in this order, we try to use commutativity rules to have first all operations by transaction 2, then by 3, and then by 1.

$$\begin{aligned} & r_1(x) r_2(x) w_2(x) r_3(x) w_1(y) w_3(x) w_1(x) \\ \sim\ & r_2(x) r_1(x) w_2(x) r_3(x) w_1(y) w_3(x) w_1(x), \end{aligned}$$

but now we are stuck because we cannot commute $r_1(x)$ with $w_2(x)$, as transaction 1 reads the old value of x .

$$\begin{aligned} & r_2(z) r_1(x) r_1(y) w_2(z) w_1(x) w_1(y) w_2(x) \\ \sim\ & r_1(x) r_2(z) r_1(y) w_2(z) w_1(x) w_1(y) w_2(x) && \text{reads always commute} \\ \sim\ & r_1(x) r_1(y) r_2(z) w_2(z) w_1(x) w_1(y) w_2(x) && \text{reads always commute} \\ \sim\ & r_1(x) r_1(y) r_2(z) w_1(x) w_2(z) w_1(y) w_2(x) && \text{writes to different pages commute} \\ \sim\ & r_1(x) r_1(y) r_2(z) w_1(x) w_1(y) w_2(z) w_2(x) && \text{writes to different pages commute} \\ \sim\ & r_1(x) r_1(y) w_1(x) r_2(z) w_1(y) w_2(z) w_2(x) && \text{reads and writes to different pages commute} \\ \sim\ & r_1(x) r_1(y) w_1(x) w_1(y) r_2(z) w_2(z) w_2(x) && \text{reads and writes to different pages commute} \end{aligned}$$

This is now a serial history, so the original history is serializable.