Mgr inż. Szymon Wojciechowski, TP Poniedziałek 18:55

DOKUMENTACJA PROJEKTOWA

Bazy danych 1

1. Deklaracja tematu

1.1. Skład grupy

- Maciej Radecki
- Andrii Zhukov
- Tomasz Górniak

1.2. Opis profilu aplikacji

Aplikacja służąca do rejestrowania czasu pracy. Możliwość wykorzystania w każdej firmie zatrudniającej Większą ilość pracowników pracujących „na godziny”. Pomocna w ewidencji czasu pracy a następnie generowaniu listy przepracowanych godzin w zadanym okresie – podstawy do wypłaty.

1.3 Opis środowiska

Użytkowników dzielimy na 2-3 grupy:

- Standardowy użytkownik – możliwość „odbicia się” rozpoczynającego i kończącego dzień pracy – tych będzie najwięcej i będą wykorzystywać system najczęściej – 2 razy dziennie
- Kadry/Płace – możliwość generowania raportu czasu pracy w zadanym okresie, kontrola ilości użytkowników „w pracy” (jest odbicie rozpoczynające zmianę, ale nie ma kończącego)
- Kierownik – uprawnienia takie jak powyżej plus możliwość dodawania / usuwania nowych pracowników (symulacja rekrutacji i zwolnień)

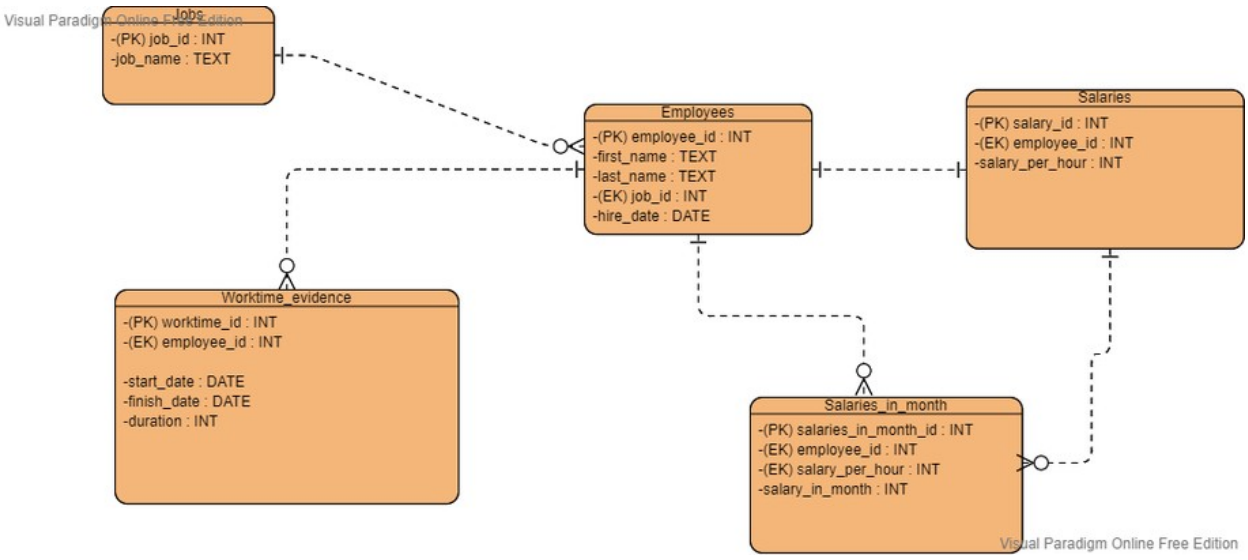
Na podstawie powyższego opisu można założyć, że dziennie będzie ok. 2*ilość_standardowych użyć systemu. Logowania kadr i kierownictwa będą pomijalnie małe ze względu na mała liczność tych grup.

1.4 Przykłady przypadków użycia systemu

- Standardowy użytkownik
  - po przyjściu do pracy „odbija się”, czyli przekazuje, kiedy rozpoczął pracę
  - przed wyjściem z pracy „odbija się”, tym razem przekazuje, kiedy zakończył pracę
  - może wygenerować rozliczenie w danym okresie
- Użytkownik kadry/płace
  - na koniec miesiąca generuje raport, dotyczący przepracowanych godzin przez pracowników
  - może sprawdzić, ile w danym dniu znajduje się pracowników w pracy
  - może przygotować rozliczenie za dany okres dla danego działu
- Menadżer
  - może zatrudnić nowego pracownika
  - może zwolnić pracownika
  - może zmienić pracownikowi pensję
  - może zmienić pracownikowi stanowisko
  - może dać pracownikowi premię

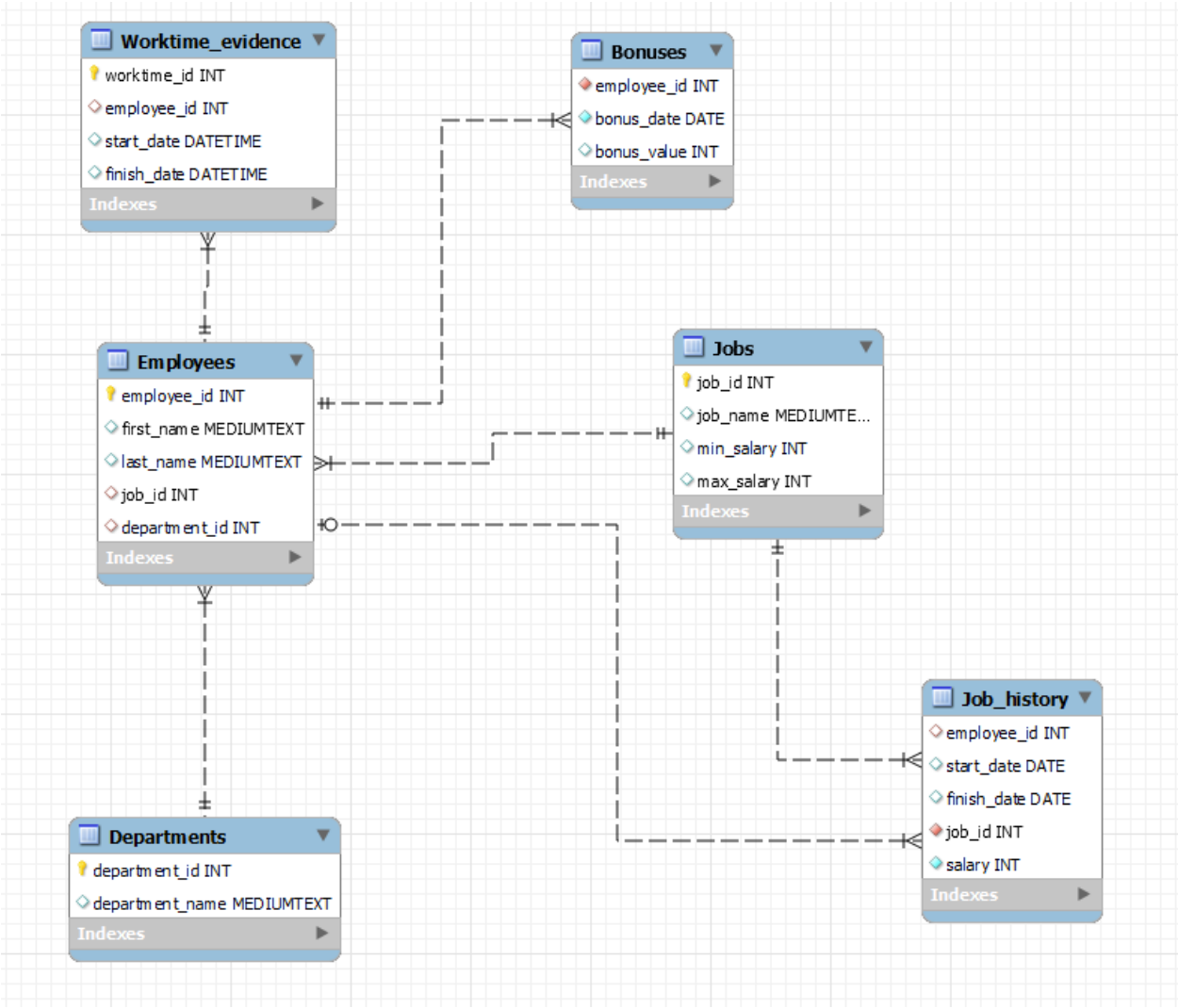
2. Diagram ER

2.1. Pierwszy diagram



Po konsultacjach z panem Wojciechowskim zostały wprowadzone zmiany w osobnych tabelach, stworzona tabela działów (Departments) oraz naliczonych bonusów (Bonuses). Tabela Jobs ma relacje typu jeden-do-wielu, natomiast Employees ma łączenia typu wiele-do-wielu. Niżej zostanie przedstawiony końcowy diagram po wprowadzonych zmianach.

2.2. Drugi diagram



3. Implementacja

3.1. Opracowanie struktury oraz danych testowych

Do wygenerowania przykładowych liczb i dat używany był system random.org. Natomiast nazwy działów i prac, imiona i nazwiska pracowników nie potrzebowały specyficznych narzędzi i były wpisane ręcznie, zostały wymyślone.

3.2. Listing przygotowanych zapytań

```
INSERT INTO Worktime_evidence (worktime_id,
employee_id, start_date, finish_date)
VALUES ('14', '21', SYSDATE(), NULL);
```

```
Use mydb;
UPDATE Worktime_evidence SET finish_date = NOW()
WHERE employee_id = “13”;
```

```
SELECT Employees.employee_id, SUM
((Worktime_evidence.finish_date –
Worktime_evidence.start_date) * 24 * Job_history.salary)
FROM (Employees INNER JOIN Worktime_evidence ON
Employees.employee_id = Worktime_evidence.employee_id)
INNER JOIN Job_history ON Employees.employee_id =
                          Job_history.employees_employee_id
WHERE Worktime_evidence.start_date >= '2000-02-10'
AND Worktime_evidence.finish_date <= '2022-06-13' AND
Worktime_evidence.start_date >= Job_history.start_date
AND                       (Worktime_evidence.finish_date <=
Job_history.finish_date OR Job_history.finish_date IS NULL)
GROUP BY Employees.employee_id;
```

```
SELECT COUNT(*) FROM Worktime_evidence
WHERE start_date < NOW() AND finish_date IS NULL;
```

```
SELECT Employees.department_id,
SUM((Worktime_evidence.finish_date -
Worktime_evidence.start_date) * 24 * Job_history.salary)
FROM (Employees INNER JOIN Worktime_evidence ON
Employees.employee_id =
Worktime_evidence.employee_id) INNER JOIN Job_history
ON Employees.employee_id =
  Job_history.employees_employee_id
WHERE Employees.department_id = 1 AND
Worktime_evidence.start_date >= '2000-02-10' AND
Worktime_evidence.finish_date <= '2022-06-13' AND
Worktime_evidence.start_date >=
  Job_history.start_date AND
(Worktime_evidence.finish_date <= Job_history.finish_date
OR Job_history.finish_date IS NULL)
GROUP BY Employees.employee_id;
```

```
Use mydb;
INSERT INTO Employees(employee_id, first_name,
last_name, job_id, department_id) VALUES (28, "Piotr",
"Kowalski", 954772, 1);
INSERT INTO Job_history(employees_employee_id,
start_date, finish_date, job_id, salary) VALUES(28,
CURDATE(), NULL, 954772, 5000);
```

```
UPDATE Job_history
        SET finish_date = CURDATE()
        WHERE employee_id == 28 AND job_id ==
954772;
```

```
UPDATE Job_history
SET finish_date = CURDATE()
WHERE employee_id == 28 AND job_id == 954772;

INSERT INTO Job_history(employee_id, start_date,
finish_date, job_id, salary)
VALUES(28, CURDATE(), NULL, 954772, 5000);
```

```
UPDATE Job_history
SET finish_date = CURDATE()
WHERE employee_id == 28 AND job_id == 954772;

INSERT INTO Job_history(employee_id, start_date,
finish_date,      job_id, salary)
VALUES(28, CURDATE(), NULL, 4522034, 5000);
```

```
INSERT INTO Bonuses(employee_id, bonus_date,
bonus_value)
VALUES("28",SYSDATE,"200");
```