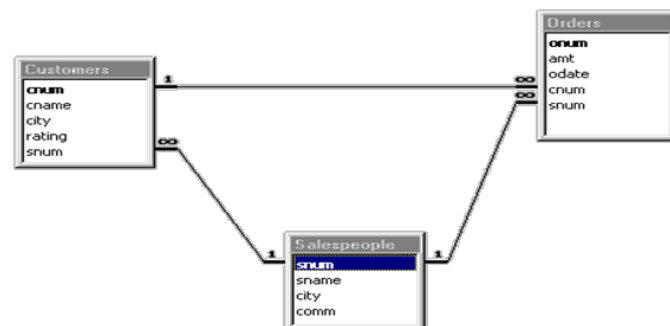


Бази даних

Частина 6 З'єднання



Навчальний курс
Валько Н.В.

SQL

Мартин Грабер

МАРТИН ГРУБЕР

Понимание SQL

Перевод Лебедева В.Н.

Под редакцией Булычева В.Н.

МОСКВА, 1993

Джеймс Р. Грофф
Пол Н. Вайнберг

SQL

Полное руководство

Второе издание,
переработанное и дополненное

Перевод с английского
под редакцией В.Р. Гинзбурга

Линн Бейли

Изучаем SQL

Приведи
в порядок
свои
отношения
с данными



Прекрати путать
термины



Освой концепцию
и синтаксис SQL
максимально
эффективно



Перестань
смушаться



Зміст

□ З'єднання

boy_id 	boy	toy_id 
1	Дэйви	3
2	Бобби	5
3	Бивет	2
4	Ричи	1

toy_id 	toy
1	обруч
2	самолет
3	солдатики
4	губная гармошка
5	бейсбольные карточки

boy	toy
Ричи	обруч
Бивет	самолет
Дэйви	солдатики
Бобби	бейсбольные карточки

Id_toy	Toy_name
--------	----------

1	М'яч
2	Йо-йо

id_t	Colour
------	--------

1	Білий
---	-------

2	Синій
---	-------

2	Зелений
---	---------

1	Синій
---	-------

З'єднання

- Join – операція з'єднання таблиць в SQL, яка сполучає дві таблиці в реляційній базі даних, утворюючи нову тимчасову таблицю, яку інколи називають «з'єднаною таблицею».

Типи з'єднання

- внутрішнє – INNER,
- зовнішнє – OUTER
 - ліве – LEFT OUTER,
 - праве – RIGHT OUTER
 - повне – FULL OUTER.
- перехресне – CROSS
- самоз'єднання (SELF-JOIN)



Синтаксис

FROM

Table1

{**INNER** | {**LEFT** | **RIGHT** | **FULL**} **OUTER** | **CROSS** }

JOIN

Table2

{**ON** <condition> | **USING** (field_name [... n])}

INNER

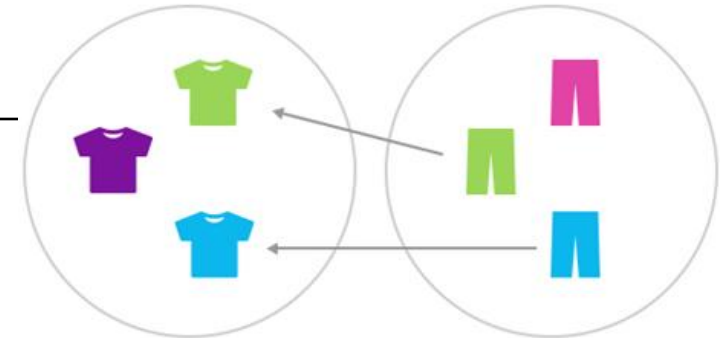
- обчислюється декартів добуток усіх записів таблиць
- усі записи таблиці А буде з'єднано з кожним із записів таблиці В, після чого в результатній таблиці залишаться лише ті записи, які задовольняють предикат з'єднання.
- Комутативне (порядок таблиць не важливий)

INNER

color_shirt	color_pants
green	green
blue	blue

INNER JOIN

according to colors



result of matching
shirts to pants



```
SELECT color_shirt, color_pants  
FROM shirt INNER JOIN pants  
ON color_shirt=color_pants;
```


OUTER

- З'єднання двох таблиць, в результат якого обов'язково входять всі рядки або однієї, або обох таблиць.
 - ліве – некомутативне (порядок таблиць важливий),
 - праве – некомутативне
 - повне – комутативне.

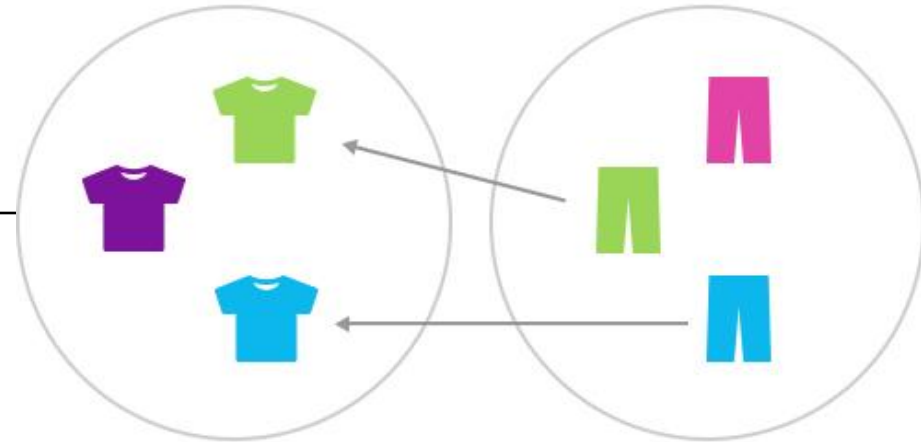


LEFT

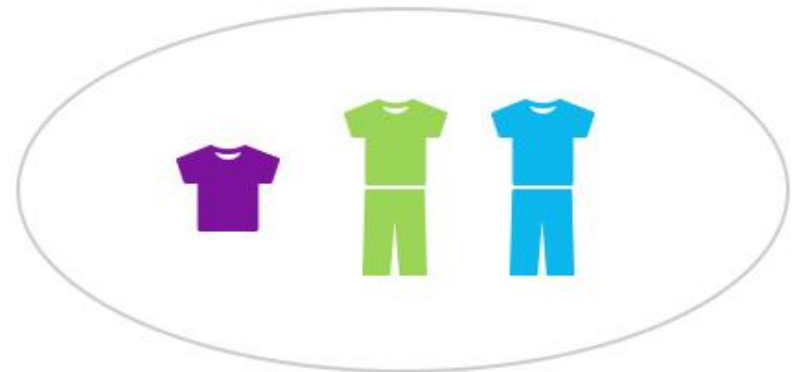
color_shirt	color_pants
yellow	NULL
green	green
blue	blue

LEFT JOIN

according to colors



result of matching
shirts to pants

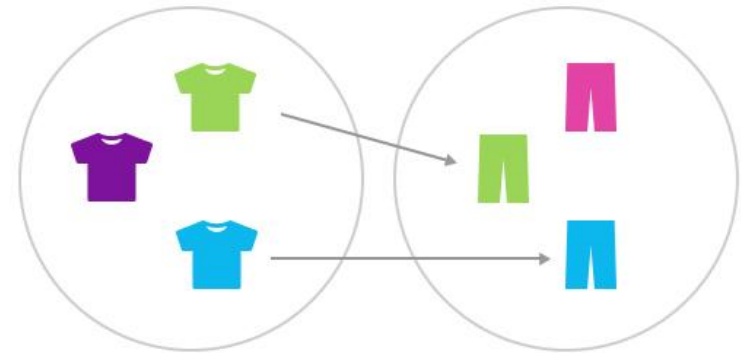


```
SELECT color_shirt, color_pants
FROM shirt LEFT JOIN pants
ON color_shirt=color_pants;
```

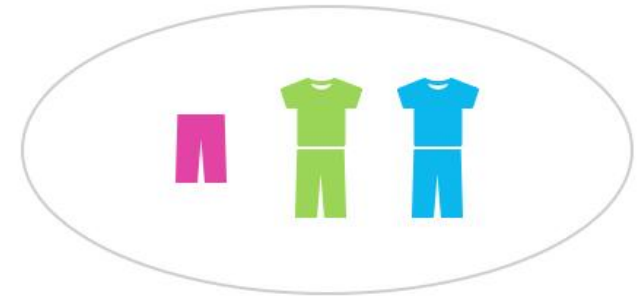
RIGHT

color_shirt	color_pants
NULL	pink
green	green
blue	blue

RIGHT JOIN
according to colors



result of matching
shirts to pants



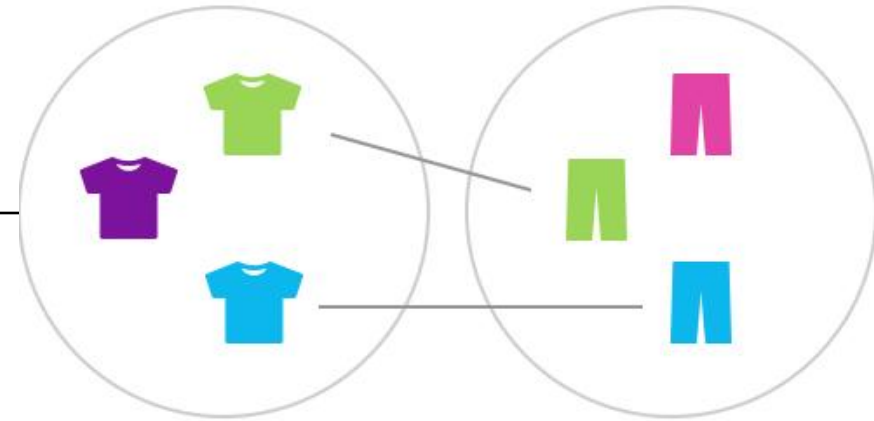
```
SELECT color_shirt, color_pants
FROM shirt RIGHT JOIN pants
ON color_shirt=color_pants;
```



FULL

color_shirt	color_pants
yellow	NULL
green	green
blue	blue
NULL	pink

FULL JOIN
according to colors



result of matching
shirts to pants

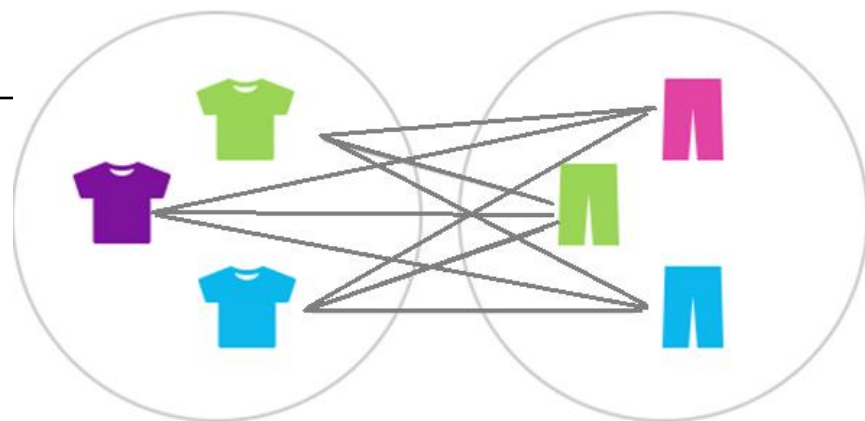


```
SELECT color_shirt, color_pants  
FROM shirt FULL JOIN pants  
ON color_shirt=color_pants;
```

CROSS JOIN

- Декартовий добуток двох таблиць
- Кожен рядок однієї таблиці з'єднується з кожним рядком другої таблиці, даючи тим самим в результаті всі можливі поєднання рядків двох таблиць

CROSS JOIN
according to colors



result of matching
shirts to pants



CROSS JOIN

```
SELECT color_shirt, color_pan
```

```
FROM shirt AS s
```

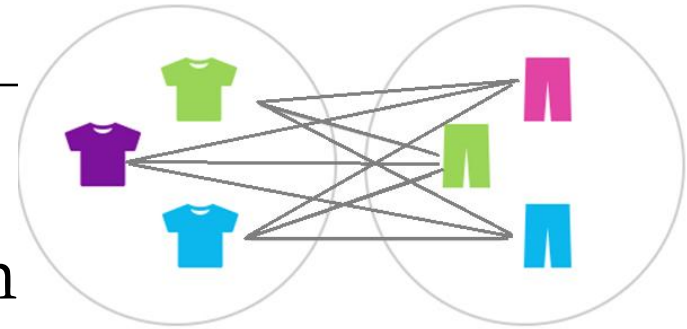
```
CROSS JOIN pan AS p;
```

```
SELECT color_shirt, color_pants
```

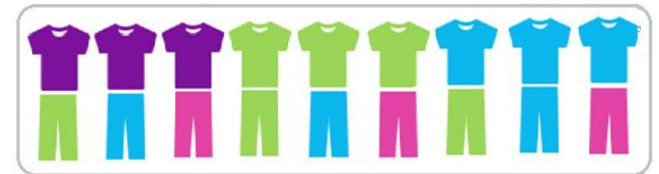
```
FROM shirt , pan ;
```

CROSS JOIN

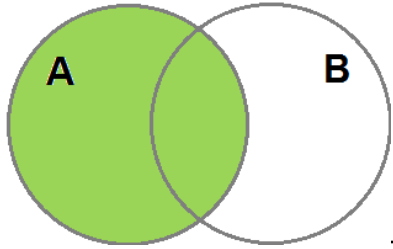
according to colors



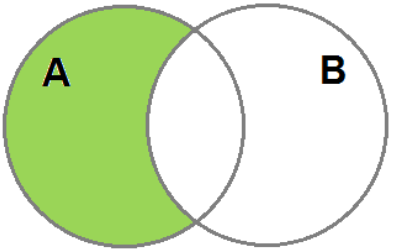
result of matching
shirts to pants



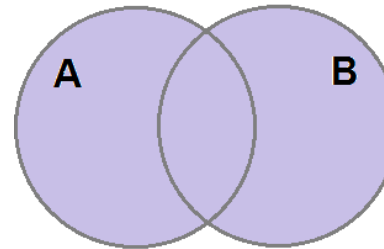
SQL JOINS



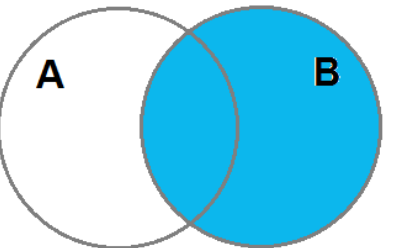
```
SELECT *  
FROM tab_A  
LEFT JOIN tab_B  
ON a.key=b.key
```



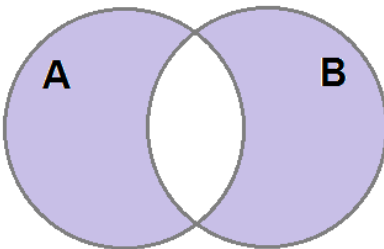
```
SELECT *  
FROM tab_A  
LEFT JOIN tab_B  
ON a.key=b.key  
WHERE b.key IS NULL
```



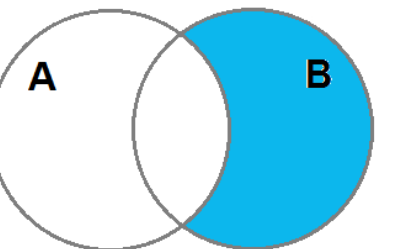
```
SELECT *  
FROM tab_A  
FULL JOIN tab_B  
ON a.key=b.key
```



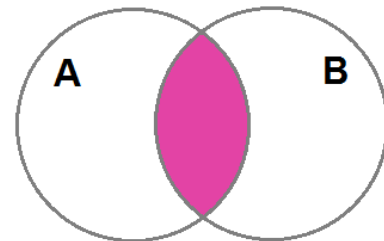
```
SELECT *  
FROM tab_A  
RIGHT JOIN tab_B  
ON a.key=b.key
```



```
SELECT *  
FROM tab_A  
FULL JOIN tab_B  
ON a.key=b.key  
WHERE a.key IS NULL  
OR b.key IS NULL
```



```
SELECT *  
FROM tab_A  
RIGHT JOIN tab_B  
ON a.key=b.key  
WHERE a.key IS NULL
```



```
SELECT *  
FROM tab_A  
INNER JOIN tab_B  
ON a.key=b.key
```

Внутрішнє з'єднання

INNER JOIN

- Внутрішнє з'єднання комбінує записи з двох таблиць у відповідності з заданою умовою

SELECT somecolumns

FROM table1

INNER JOIN table 2

ON somecondtion ;

WHERE somecondtion



Внутрішнє з'єднання INNER JOIN

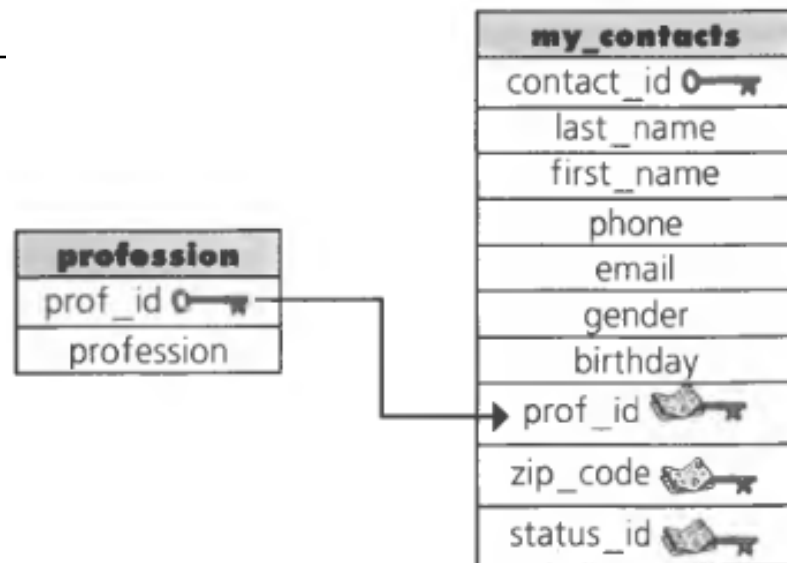
SELECT

mc.last_name,
mc.first_name,
p.profession

FROM my_contacts AS mc

INNER JOIN profession AS p

ON mc.contact_id = p.prof_id



за умови, що значення стовпця «contact_id» таблиці my_contacts збігається зі значенням стовпця «prof_id» таблиці profession

Еквівалентне з'єднання INNER JOIN

```
SELECT boys.boy , toys.toy  
FROM boys  
INNER JOIN toys  
ON boys.toy_id = toys.toy_id ;
```

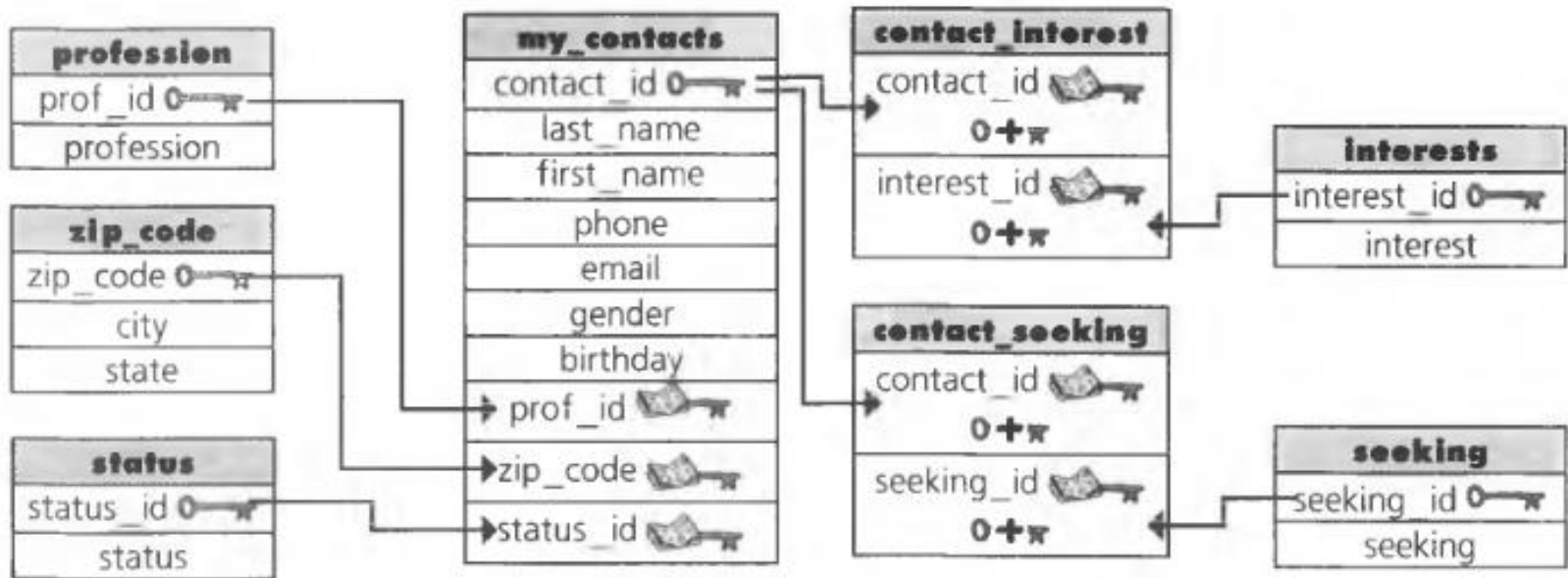
boy_id 🔑	boy	toy_id 🔑
1	Дэйви	3
2	Бобби	5
3	Бивет	2
4	Ричи	1

toy_id 🔑	toy
1	обруч
2	самолет
3	солдатики
4	губная гармошка
5	бейбольные карточки

boy	toy
Ричи	обруч
Бивет	самолет
Дэйви	солдатики
Бобби	бейбольные карточки

```
SELECT mc.email, p.profession
FROM my_contacts mc
INNER JOIN profession p
ON mc.prof_id = p.prof_id
```

Запит, який повертає адреси електронної пошти (email) і професії (profession) кожної людини в my_contacts

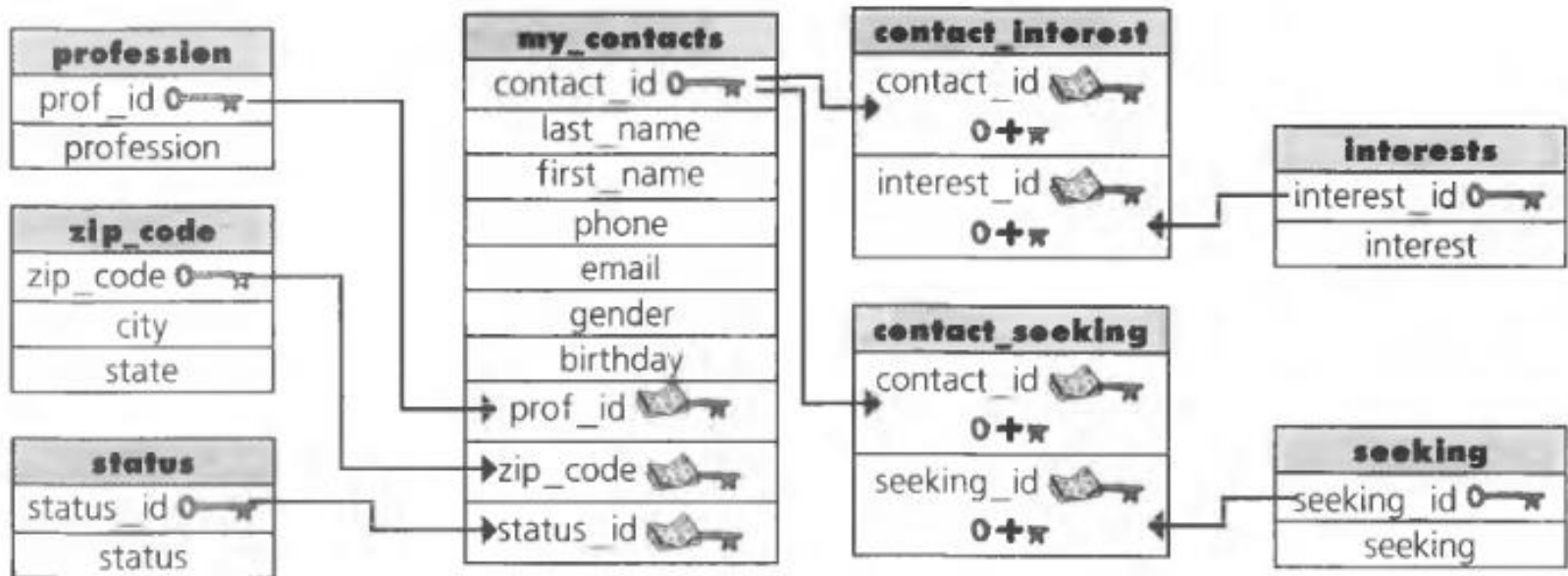


```

SELECT mc.first_name,
       mc.last_name, z.state
FROM my_contacts mc
INNER JOIN zipjcode z
ON mc.zip_code = z.zip_code;

```

Запит, який повертає ім'я
(first_name), прізвище (last_name) і
штат (state) кожної людини в
my_contacts

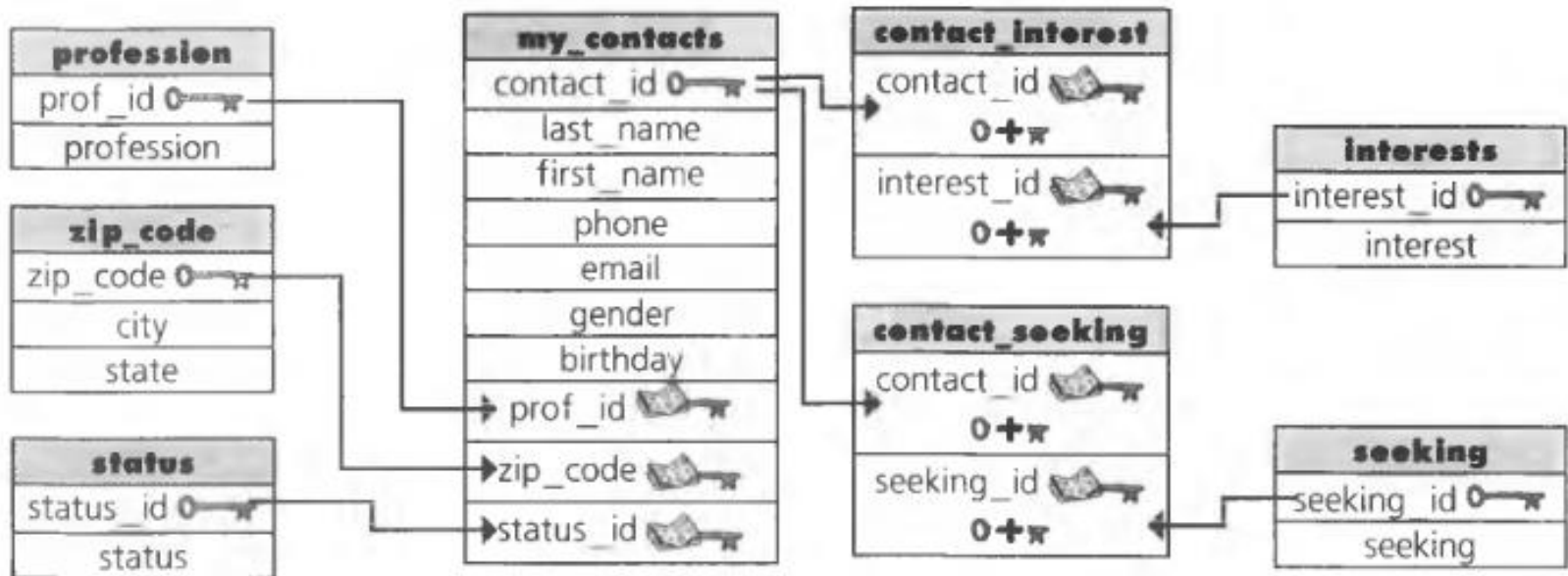


```

SELECT mc.first_name,
       mc.last_name, s.status
FROM my_contacts mc
INNER JOIN status s
ON mc.status_id = s.status_id;

```

Запит, який повертає ім'я (firstname), прізвище (lastname) і сімейний стан (status) кожної людини в my_contacts



Нееквівалентне з'єднання

Нееквівалентне з'єднання повертає записи,
у яких задані значення стовпців НЕ рівні

```
SELECT boys.boy , toys.toy
```

```
FROM boys
```

```
INNER JOIN toys
```

```
ON boys.toy_id < > toys.toy_id
```

```
ORDER BY boys.boy
```

boys

boy_id	boy	toy_id
1	Дэйви	3
2	Бобби	5
3	Бивер	2
4	Ричи	1

toys

toy_id	toy
1	обруч
2	самолет
3	солдатики
4	губная гармошка
5	бейсбольные карточки

boy	toy
Бивер	обруч
Бивер	солдатики
Бивер	губная гармошка
Бивер	бейсбольные карточки
Бобби	солдатики
Бобби	губная гармошка
Бобби	обруч
Бобби	самолет
Дэйви	обруч
Дэйви	самолет
Дэйви	губная гармошка
Дэйви	бейсбольные карточки
Ричи	самолет
Ричи	солдатики
Ричи	губная гармошка
Ричи	бейсбольные карточки

Природне з'єднання

Природне з'єднання можливе тільки в тому випадку, якщо стовпець, по якому виконується з'єднання, має однакові імена в обох таблицях.

```
SELECT boys.boy , toys.toy
FROM boys
NATURAL JOIN toys
```

boys

boy_id	boy	toy_id
1	Дэйви	3
2	Бобби	5
3	Бивер	2
4	Ричи	1

toys

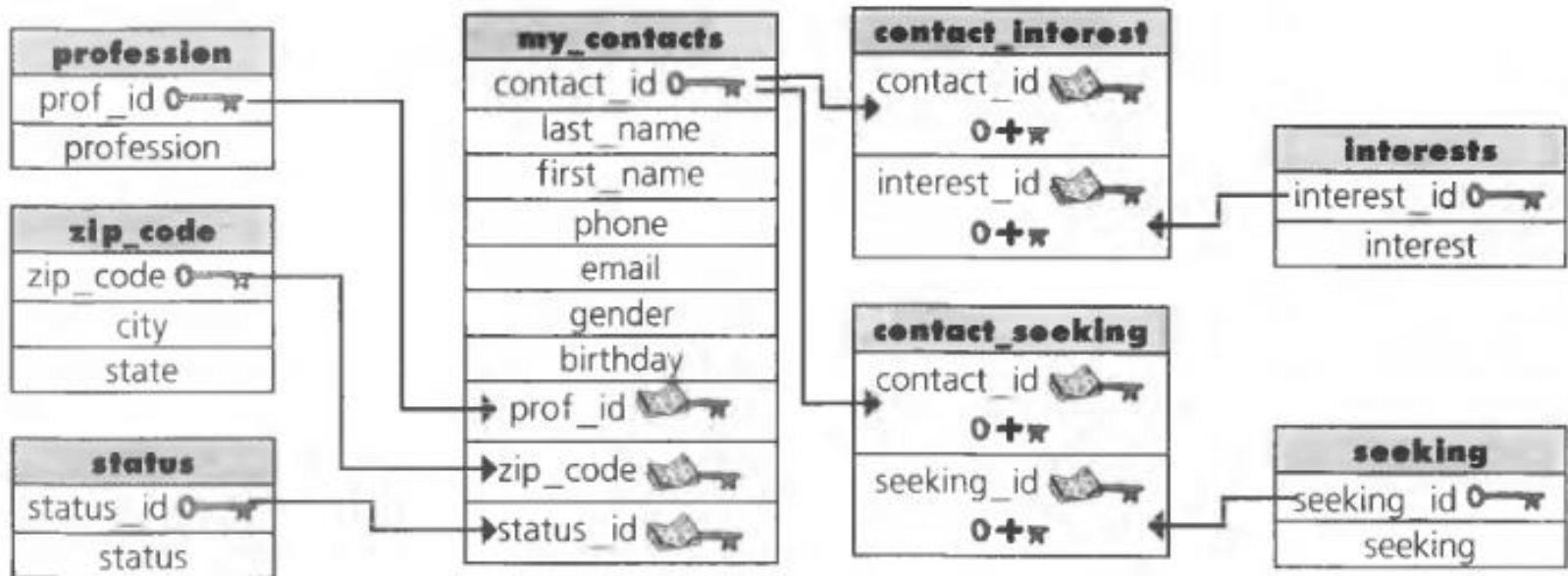
toy_id	toy
1	обруч
2	самолет
3	солдатики
4	губная гармошка
5	бейсбольные карточки

boy	toy
Ричи	обруч
Бивер	самолет
Дэйви	солдатики
Бобби	губная гармошка

```
SELECT mc.email , p.profession
FROM my_contacts mc
INNER JOIN profession p;
```

Запит, який повертає адреси електронної пошти (email) і професії (profession) кожної людини в my_contacts.

Умова ON не обов'язково тому що імена зовнішнього і первинного ключів співпадають

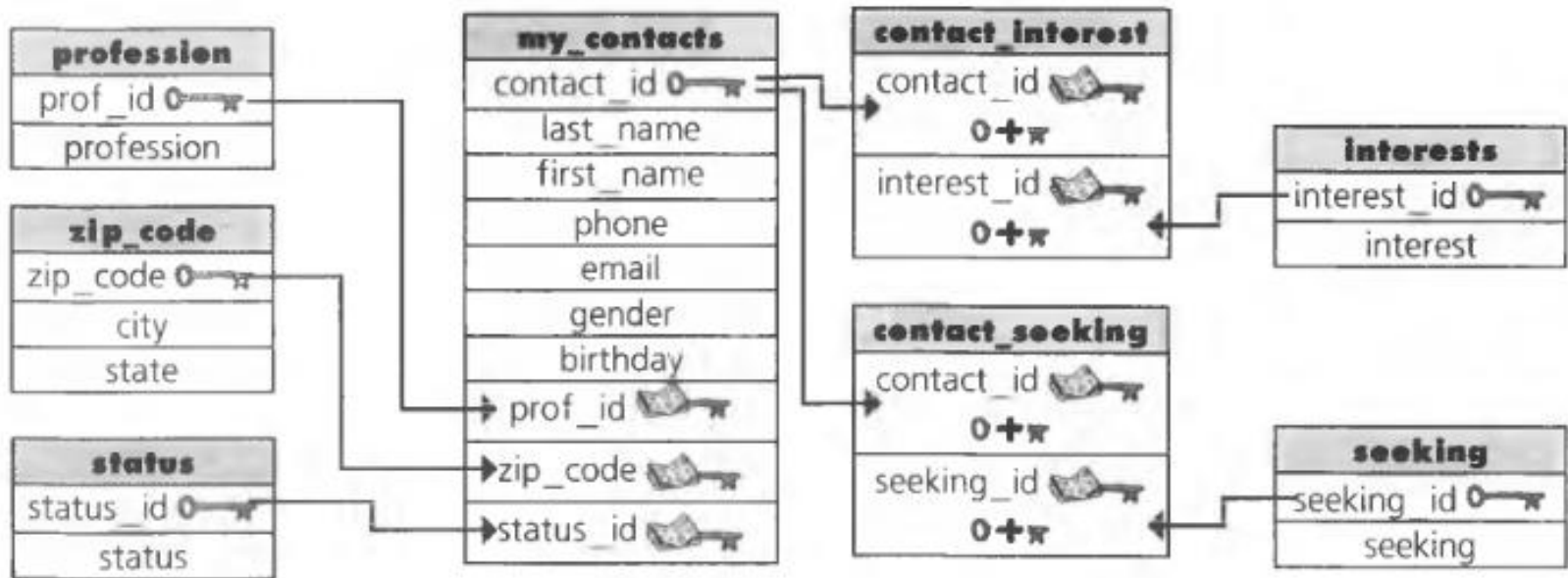



```

SELECT mc.first_name,
       mc.last_name, s.status
FROM my_contacts mc
INNER JOIN status s
ON mc.status_id <> s.status_id;

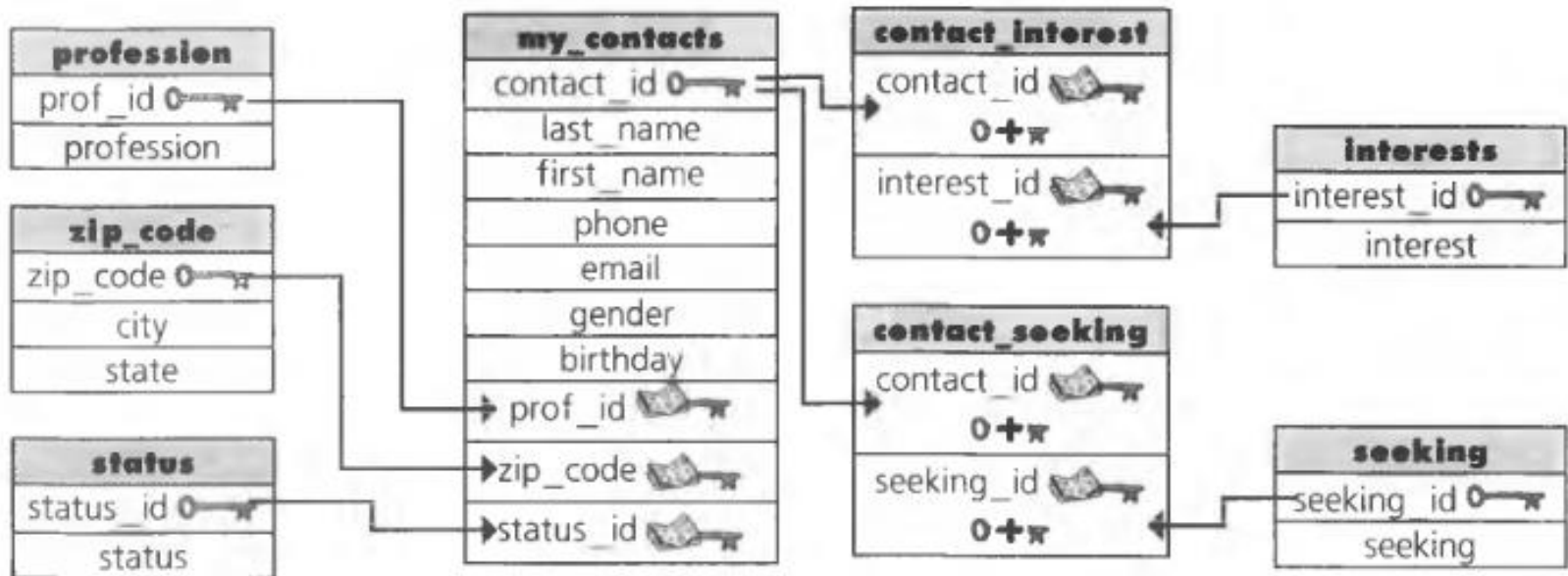
```

Запит, який повертає ім'я (firstname), прізвище (lastname) і сімейний стан (status), яким не володіє кожна людина в my_contacts.



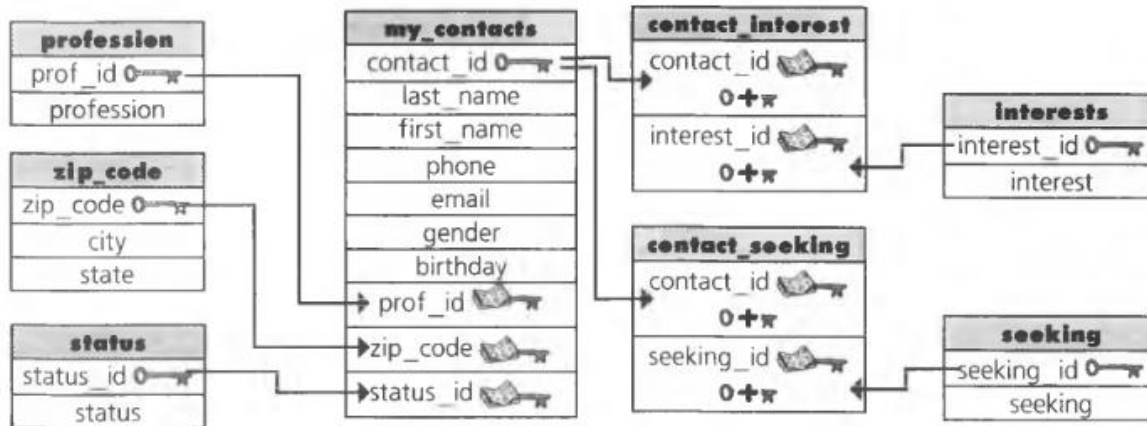
```
SELECT mc.first_name,  
       mc.last_name, z.state  
FROM my_contacts mc  
INNER JOIN zip_code z;
```

Запит, який повертає ім'я
(first_name), прізвище
(last_name) і штат (state) кожної
людини в my_contacts.



```
SELECT mc.first_name mc.last_name, ci.interest_id
FROM my_contacts me
INNER JOIN contact_interest ci
ON me.contact_id = ci.contact_id
```

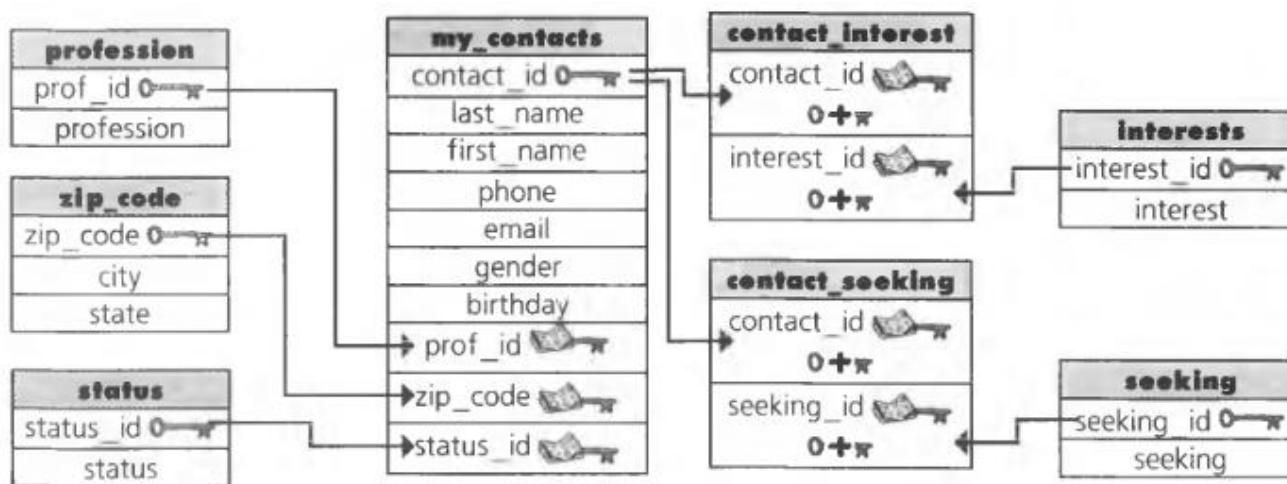
```
SELECT mc.first_name mc.last_name, ci.interest_id
FROM my_contacts me
NATURAL JOIN contact_interest ci;
```



запити з різними
з'єднаннями для
отримання парних
записів з таблиць
my_contacts і
contact_interest

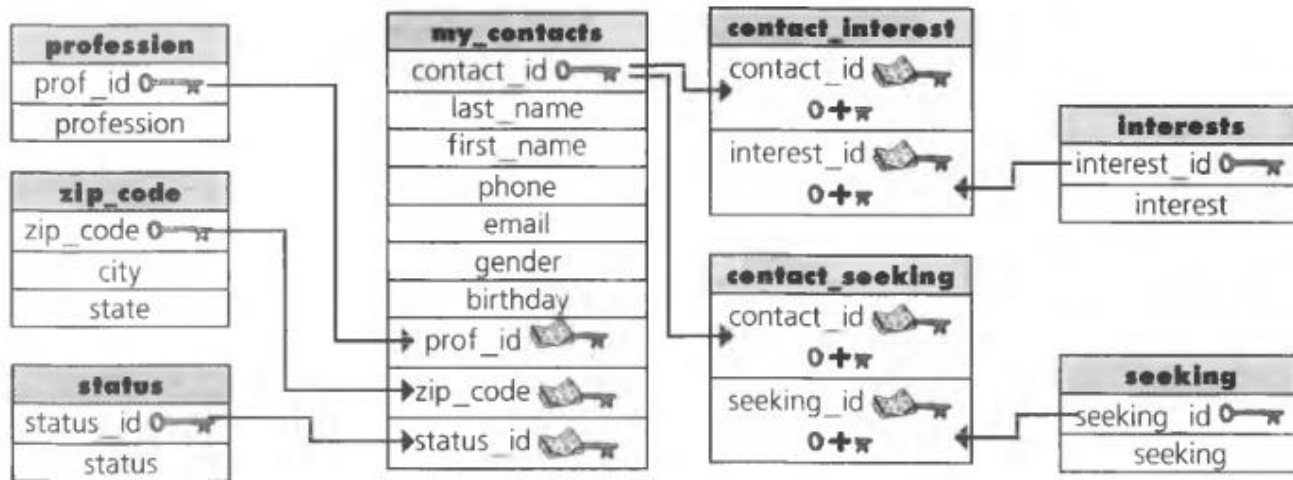
```
SELECT * FROM contact_seeking  
CROSS JOIN seeking;
```


запит для
отримання всіх
можливих
комбінацій записів
з таблиць
contact_seeking і
seeking



```
SELECT p.profession
FROM my_contacts mc
INNER JOIN profession p
ON mc.prof_id = p.prof_id
GROUP BY profession
ORDER BY profession;
```

список професій людей з
таблиці my contacts, але
без дублікатів і в
алфавітному порядку





```
SELECT mc.last_name, mc.first_name, mc.phone  
FROM my_contacts AS mc  
NATURAL JOIN job_desired AS jd  
WHERE jd.title = 'Веб-розробник'  
AND jd.salary_low < 105 000;
```

Контакти людей, які шукають роботу веб-розробника, готові працювати за запропоновану суму.

За одну «salary_low» ми переконуємося в тому, що запропонована зарплата не нижче запитуваної мінімуму.



Підзапит

З'єднання

```
SELECT last_name, first_name  
FROM my_contacts mc  
NATURAL JOIN zip_code zc  
WHERE zc.city = 'Мемфіс' AND zc.state = 'TN'
```

Підзапит

```
SELECT last_name, first_name  
FROM my_contacts  
WHERE zip_code = (SELECT zip_code  
FROM zip_code  
WHERE city='Мемфіс' AND state='TN' );
```

Вибираємо з my_contacts імена людей, що живуть в Мемфісі (штат Теннессі).

Підзапит

Знайти людей, посади яких збігаються з однією із запропонованих вакансій

```
SELECT mc.last_name, mc.first_name, mc.phone , jc.title  
FROM job_current AS jc  
NATURAL JOIN my_contacts AS mc  
WHERE  
jc.title IN ('Веб-дизайнер', 'Веб-разработчик', 'Официант');
```

mc.first_name	mc.last_name	mc.phone	jc.title
Джо	Лонниган	(555) 555-3214	Повар
Венди	Хиллерман	(555) 555-8976	Официант
Шон	Миллер	(555) 555-4443	Веб-дизайнер
Джаред	Колуэй	(555) 555-5674	Веб-разработчик
Хуан	Гарза	(555) 555-0098	Веб-разработчик

Подзапрос

Знайти людей, посади яких збігаються з однією із запропонованих вакансій

```
SELECT mc.first_name, mc.last_name, mc.phone, jc.title  
FROM job_current AS jc  
NATURAL JOIN my_contacts AS mc  
WHERE  
    jb.title IN (SELECT title FROM job_listing) ;
```

Второй запрос, в котором извлекаются совпадения из таблицы professions, мы назовем **ВНЕШНИМ** запросом, потому что в него «упакован» другой, **ВНУТРЕННИЙ** запрос: все профессии из списка в скобках были получены в результате **первого** запроса — того, который выбирал все вакантные должности из таблицы job_current.



Подзапрос

```
SELECT  some_column , another_column  
FROM    table  
WHERE   column = (SELECT  column FROM table);
```

- Підзапити можуть використовуватись в командах INSERT, DELETE, UPDATE і SELECT.
- Найскладніше в підзапитах - визначення того, яку частину запиту слід перетворити в підзапит (і чи потрібно це робити взагалі).

Підзапит

Аналіз питання:

У кого зі співробітників найбільша зарплата?

- Обираємо максимальну зарплату:

```
SELECT MAX(salary) FROM job_current
```

- Обираємо співробітників:

```
SELECT mc.first_name, mc.last_name FROM my_contacts AS mc
```

- Дані про заробітки людей:

```
SELECT mc.first_name, mc.last_name, jc.salary  
FROM my_contacts AS mc NATURAL JOIN job_current AS jc
```

- Додаємо умову WHERE:

```
SELECT mc.first_name, mc.last_name, jc.salary  
FROM my_contacts AS mc NATURAL JOIN job_current AS jc  
WHERE jc.salary = (SELECT MAX(salary) FROM job_current)
```

Підзапит

Інформацію про всіх веб-розробників і про те, наскільки більше (або менше) середнього рівня вони заробляють.

```
SELECT mc.first_name, mc.last_name, jc.salary,  
(jc.salary – SELECT AVG(salary)  
FROM job_current  
WHERE title = 'Веб-разработчик')
```

Запит для визначення
середньої зарплати
веб-розробників в
таблиці job_current

```
FROM my_contacts AS mc NATURAL JOIN job_current AS jc  
WHERE jc.title = 'Веб-разработчик'
```

Підзапит з природним з'єднанням

Вывод данных только тех людей,
у которых зарплата выше чем у конкретного человека

```
SELECT mc.first_name, mc.last_name, jc.salary
FROM my_contacts AS mc NATURAL JOIN job_current AS jc
WHERE jc.salary > (SELECT jc.salary
                   FROM my_contacts me
                   NATURAL JOIN job_current jc
                   WHERE email = 'andy@weather.com')
```

Запрос на зарплату
сотрудника с почтой
andy@weather.com

Подзапрос как столбец SELECT

```
SELECT mc.first_name, mc.last_name,  
      (SELECT state FROM zip_code  
       WHERE mc.zip_code = zip_code) AS state  
FROM my_contacts me;
```

Перебрать все записи в таблице my_contacts. Для каждой записи получить имя, фамилию и штат (для чего запрос получает почтовый индекс и сопоставляет его с обозначением штата по таблице zip_code).

mc.first_name	mc.last_name	state
Джо	Лонниган	TX
Венди	Хиллерман	CA
Шон	Миллер	NY
Джаред	Колуэй	NJ
Хуан	Гарза	CA

Подзапрос с несколькими значениями: IN, NOT IN

```
SELECT mc.first_name, mc.last_name, mc.phone, jc.title  
FROM job_current AS jc  
NATURAL JOIN my_contacts AS mc  
WHERE  
    jb.title IN (SELECT title FROM job_listing);
```



NOT IN



Питання
