Зміст

1. Постановка задачі

Інформаційна система здійснює підтримку діяльності МОЗ. ІС повинна здійснювати: ведення списку жителів (ПІБ, стать, дата народження, ...) і ведення списку адрес (вулиці і будинки: кожен будинок має свій унікальний номер), відомості про прописку.

Типовими для інформаційної системи є питання:

• Надання відомостей про мешканців певної квартири (будинку).

• Список жителів, які ніде не зареєстровані.

Під час аналізу вимог до ІС будо встановлене бізнес-правило – одна людина мешкає за однією адресою. Кожна вулиця має лише одну назву.

Розширене завдання

Під час експлуатації ІС виявились наступні особливості: людина може змінювати місце прописки; вулиці змінюють назви, треба забезпечити пошук із урахуванням старих і нових назв.

Додатковий функціонал: за кожним з будинків закріплена лікарня.

Реалізуйте запит:

• Список адрес на певній вулиці, по яких ніхто не зареєстрований.

• Скільки пацієнтів закріплено за кожною з лікарень?

Проаналізуйте план виконання будь-якого із запитів.

Повне завдання

Спроектуйте побудовану схему БД на метамодель (по А. Тенцеру). Перенесіть запити з базового завдання на метамодель.

2. Проектування інформаційної системи

2.1. Побудова діаграми потоків даних

Розробляється інформаційна система (ІС) призначена для підтримки діяльності МОЗ. Основні користувачі системи - співробітники лікарень. За допомогою цієї системи вони можуть отримувати різні відомості про громадян і про їх місцях проживання за допомогою запитів. Також користувачами ІС можуть бути оператори різних структур, які уповноважені вносити зміни в дані. Ці процеси можна відобразити на DF-діаграмах (див. Рис 2.1 і 2.2).

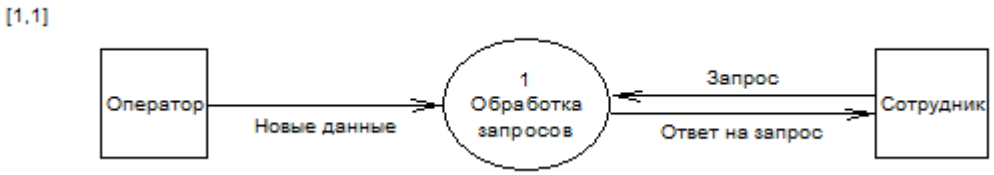

Рисунок 2.1. – DFD 0-го рівня для ІС МВС

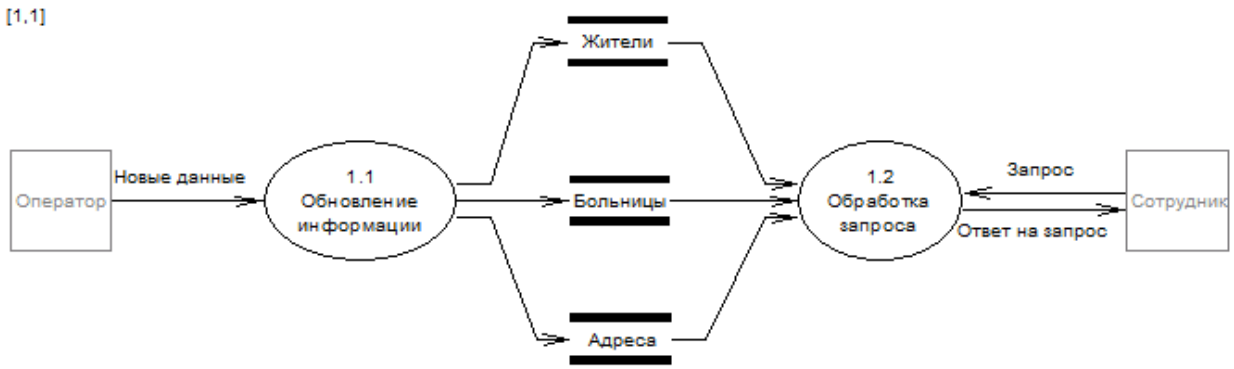

Рисунок 2.2. – DFD 1-го рівня процессу «Обработка запросов»

Для обробки процесів концептуальна модель на рис. 2.2 має наступні сховища даних: «Жителі», «Адреси» і «Лікарні». Всі запити від користувачів діляться на процес «Оновлення інформації», який вносить зміни в сховища даних і процес «Обробки запиту», який обробляє запити від співробітника, знаходить необхідну інформацію в сховищах даних і формує відповідь на запит.

2.2. Проектування бази даних

Для усунення потенційної суперечливості і надмірності даних у відносинах, виявлених на етапі побудови концептуальної моделі («Жителі», «Адреси» і «Лікарні») приведемо їх до третьої нормальної форми. Зобразимо отримані відносини і їх зв'язки на ER-діаграмі (див. рис. 2.3).

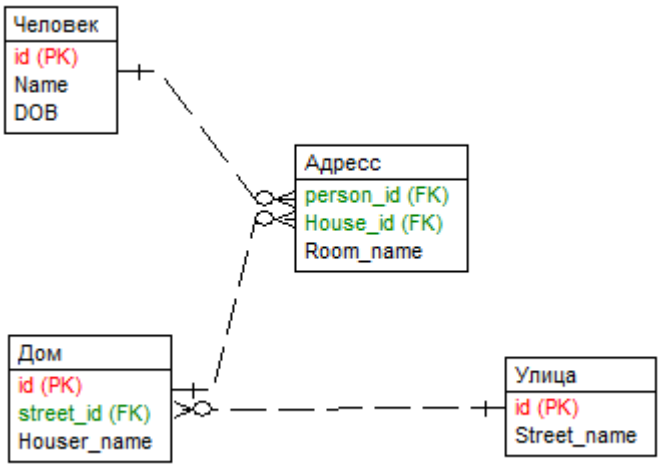

Рисунок 2.3. – ERD для ІС МОЗ

Для приведення відносин до 3НФ розіб'ємо сутність «Адреси» на сутності «Житло» і «Квартири», а також для усунення неоднозначності зв'язку багато-до-багатьох суті «Жителі» і «Квартири» введемо нову сутність «Прописка».

Відношенню «Людина» відповідає повна ФЗ: pid → last_name, first_name, mid_name, sex, birthday, passport, inn, birth_cert.

...

Під час аналізу вимог до ІС будо встановленні наступні бізнес-правила. “Одна людина мешкає за однією адресою”. Вказане гарантується альтернативним ключем (person_id, house_id) у відношенні Адрес. Правило “Кожна вулиця має лише одну назву” забезпечується тим, що назва вулиці функціонально залежить від первинного ключа у відношенні «вулиці». Вказана структура дозволяє виконувати перейменування вулиць без зміни інших даних. Додатково на назву вулиці накладено обмеження унікальності.

2.3. Логична реалізація бази даних

Проаналізувавши сутність, використовувані в моделі ІС, перейдемо до реалізації структури БД. Для цього представимо імена необхідних таблиць, атрибутів, типів, їх призначення та обмеження (див. табл. 2.1).

Таблиця 2.1 Структура БД

Таблиця	Поле	Зміст	Тип	Ключі	Обмеження
people	pid	Номер жителя	INTEGER	PK	Не пустий
	name	Фамилия жителя	VARCHAR2(100)		Не пустий
	sex	Пол	NUMBER(1,0)		Не пустий
	birthday	Дата рождения	DATE		Не пустий
…	…	…	…	…	…

3. Фізична реалізація

3.1. Фізична реалізація ІС МОЗ

Сценарій створення структури реляційної БД:

```
Create table people (
     id Integer NOT NULL ,
     name Varchar2(100) NOT NULL ,
     sex char(1) NOT NULL,
     birthday Date NOT NULL ,
     primary key ("pid"),
     CHECK sex in ('M','F')
) ;
…
```

3.2. Реалізація типових запитів

Виконаємо типові для реалізованої ІС запити:

• Надання відомостей про мешканців певної квартири (будинку).

```
select name, sex, birthday, passport
from people p
join registration r on (p.id = r.people_id)
join house h on (h.id = r.house_id)
join street s on (s.id= h.street_id)
 where s.name='Харьковская' and h.name='3/1' and
r.room_name='123';
```

name	sex	birthday
Коломиец Наталья	F	27.09.02
Коломиец Татьяна	F	13.07.89
Коломиец Роман	M	09.12.50

…

4. Розширене завдання

4.1. Міграція з прототипа на нову версію ІС

Під час експлуатації ІС виявились наступні особливості: людина може змінювати місце прописки; вулиці змінюють назви, треба забезпечити пошук із урахуванням старих і нових назв.

Додатковий функціонал: за кожним з будинків закріплена лікарня.

Для підтримки нового функціонала будуть додані нові таблиці. Представимо їх на ER-диаграмі (див рис 4.1)

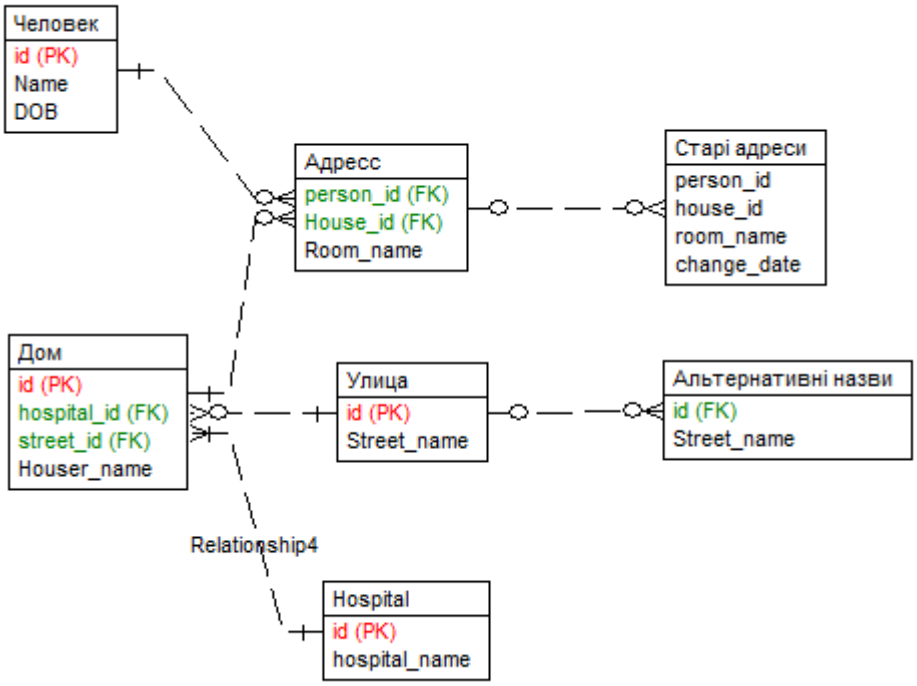

Рисунок 4.1 – ERD для ІС МОЗ після міграції

Нові вимоги передбачають збереження відомостей щодо старих назв вулиць та історичних відомостей про прописку мешканців. А також інформацію щодо приналежності будинка до лікарні.

Міграція даних буде проходити наступним чином.

1. На основі даних щодо прописки створимо таблицю з історичними даними. Цілісність даних буде забезпечуватись тригером, який буде заповнювати таблицю при зміні у таблиці з прописками.
2. На основі даних про вулиці створимо таблицю з альтернативними назвати. Цілісність даних буде забезпечуватись тригером, який буде заповнювати таблицю при зміні у таблиці з прописками. Окрім того персонал зможе додавати неформальні назви вулиць у ручному режимі.
3. Створимо таблицю із лікарнями. За заповнення цієї таблиці відповідає персонал.
4. Додамо поле «лікарня» у таблицю «Дома». Встановимо обмеження зовнішнього ключа на це поле. За заповнення цього поля відповідає персонал відповідає персонал.

```
-- Step 1
Create table registration_history AS
select r.*, SYSDATE as changed_at from registration r;
Create trigger …

-- Step 2
…
```

У наслідок зміни структури сховища треба внести корективи у базові запити, а саме:

Надання відомостей про мешканців певної квартири (будинку). Необхідно врахувати альтернативні і старі назви вулиць.

```
select name, sex, birthday, passport
from people p
join registration r on (p.id = r.people_id)
join house h on (h.id = r.house_id)
join street_alternates s on (s.id= h.street_id)
 where s.name='Харьковская' and h.name='3/1' and
r.room_name='123';
```

name	sex	birthday
Коломиец Наталья	F	27.09.02

Коломиец Татьяна	F	13.07.89
Коломиец Роман	M	09.12.50

...

4.2. Реалізація типових запитів

...

4.3. Аналіз швидкодії запитів

Проаналізуємо план виконання останнього запиту за допомогою команди set autotrace on.

```
-----------------------------------------------------------------------------------------------
| Id  | Operation                      | Name         | Rows  | Bytes | Cost (%CPU)| Time     |
-----------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT               |              |     3 |   177 |     3   (0)| 00:00:01 |
|   1 |  SORT AGGREGATE                |              |     1 |     9 |            |          |
|   2 |   MERGE JOIN                   |              |    15 |   135 |     4  (25)| 00:00:01 |
|*  3 |    TABLE ACCESS BY INDEX ROWID| adds         |     3 |    18 |     2   (0)| 00:00:01 |
|   4 |     INDEX FULL SCAN            | SYS_C008335 |     9 |       |     1   (0)| 00:00:01 |
|*  5 |    SORT JOIN                   |              |    40 |   120 |     2  (50)| 00:00:01 |
|   6 |     INDEX FULL SCAN            | SYS_C008341 |    40 |   120 |     1   (0)| 00:00:01 |
|   7 |  TABLE ACCESS FULL             | hospital     |     3 |   177 |     3   (0)| 00:00:01 |
-----------------------------------------------------------------------------------------------
```

Після виконання запиту ми отримуємо також інформацію про те, як саме SQL * Plus виконує запит, яким чином він переходить до таблиць і використовує індекси для доступу до даних. Також ми можемо побачити ієрархію виконання підзапитів, витрати процесорного і реального часу обробки кожного підзапиту і кількість обробленої інформації. В даному запиті виконується join двох таблиць "registration" і "adds" по колонці "aid". При цьому в таблиці "adds" стовпець "aid" є проіндексованим, так як він є первинним ключем. За планом виконання запиту видно, що в таблиці "registration" пошук по стовпчику "aid" виконується повністю, тому для підвищення ефективності виконання даного запиту можна додати індекс на даний стовпець.

План виконання запиту, після додавання індексу на стовпець "aid" таблиці "registration":

```
-------------------------------------------------------------------------------
```

```
| Id  | Operation           | Name     | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------
|   0 | SELECT STATEMENT    |          |     3 |   177 |     3   (0)| 00:00:01 |
|   1 |  SORT AGGREGATE     |          |     1 |     9 |            |          |
|   2 |   NESTED LOOPS      |          |    15 |   135 |     3   (0)| 00:00:01 |
|*  3 |    TABLE ACCESS FULL| adds     |     3 |    18 |     3   (0)| 00:00:01 |
|*  4 |    INDEX RANGE SCAN | REG_IDX  |     5 |    15 |     0   (0)| 00:00:01 |
|   5 |  TABLE ACCESS FULL  | hospital |     3 |   177 |     3   (0)| 00:00:01 |
-------------------------------------------------------------------------------
```

5. Застосування мета моделі при реалізації ІС МОЗ

5.1. Відображення реляційної моделі на мета модель

Класичною методикою проектування баз даних (БД) є створення окремої таблиці для кожної описуваної моделлю даних суті, потім - в процесі нормалізації - виділення окремих таблиць для зберігання атрибутів сутності (таблиці-довідники). Такий підхід добре працює для БД з відносно невеликою кількістю описуваних об'єктів (десятки) і при нескладних та статичних зв'язках між ними. Однак будь-яка зміна структури збережених даних призводить до внесення змін в структуру таблиць, ці дані відображають. Нескладна на етапі розробки, ця операція стає вкрай проблематичною при великих обсягах даних і при відсутності у розробника безпосереднього доступу до БД (наприклад, якщо вона знаходиться у замовника). Багатьом, напевно, знайомі громіздкі, повільні і вимагають величезного дискового простору операції з конвертування БД при переході на нову версію продукту. Не менш неприємна робота з БД, історично розрослася до сотень таблиць, структуру якої складно навіть зобразити в читабельному вигляді.

Одним з основоположників метамоделі став Анатолій Тенцер. Він описав п'ять основних тез, на яких повинна будуватися проектована база даних:

- Кожна сутність, інформація про яку зберігається в БД, - це об'єкт.
- Кожен об'єкт є унікальним у межах БД і має унікальний ідентифікатор.
- Об'єкт має властивості (строкові, числові, тимчасові, перелічуваних), які описують атрибути сутності.
- Об'єкти можуть бути пов'язані між собою довільним чином. Зв'язок характеризується пов'язаними об'єктами і типом зв'язку. Наприклад, співробітник фірми може бути пов'язаний з відділом, в якому він

працює, зв'язком типу «співробітник у відділі» і т.п. Зв'язок в певному сенсі аналогічна поняттю посилання на таблицю-довідник в традиційній моделі БД.

- Об'єкт може бути сховищем. У цьому випадку допускається зберігання в ньому інших об'єктів (наприклад, товару на складі).

Така БД не прив'язана ні до якої бізнес-моделі і дозволяє реалізувати «над собою» практично будь-яку бізнес-логіку. Логіка виділяється в окремий програмний шар і, як правило, реалізується на сервері додатків, де за запитом клієнта створюються об'єкти, які завантажили інформацію про себе з БД і реалізують «поведінку» об'єктів реального світу. У той же час, в силу одноманітності моделі зберігання, ці об'єкти досить легко створюються на основі базових класів, інкапсулюючих функціональність по завантаженню та збереженню властивостей і зав’язків в БД.

5.2. Фізична реалізація сховища для метамоделі

Згідно описаним вище умовам, спроектуємо концептуальну модель, що складається з 4 сутностей: Object_types, Objects, Attributes і Params. Побудуємо ER-діаграму цієї моделі (див. рис. 5.1).

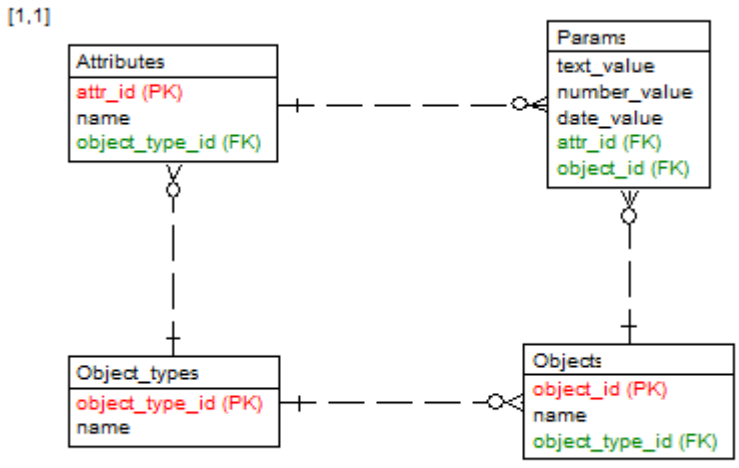

Рисунок 5.1 – ERD метамодели для ІС МОЗ.

Відносини Object_types, Objects, Attributes і Params, виявлені на етапі побудови концептуальної моделі характеризуються такими атрибутами, розглянутими в таблицях 5.1, 5.2, 5.3, 5.4.

Таблица 5.1 Атрибути сутності «Object_types»

Атрибут	Опис
object_type_id	Номер типа, первичный ключ
name	Название типа

Таблица 5.2 Атрибути сутності «Objects»

Атрибут	Опис
object_id	Номер объекта, первичный ключ
name	Название объекта
object_type_id	Внешний ключ отношения “Object_types”

Таблица 5.3 Атрибути сутності «Attributes»

Атрибут	Опис
attr_id	Номер атрибута, первичный ключ
name	Название атрибута
object_type_id	Внешний ключ отношения “Object_types”

Таблица 5.4 Атрибути сутності «Params»

Атрибут	Опис
text_value	Текстовое значение параметра
number_value	Числовое значение параметра
date_value	Значение даты параметра
attr_id	Внешний ключ отношения “Attributes”
object_id	Внешний ключ отношения “Objects”

Реалізуємо таблиці на основі сутностей та їх атрибутів (див. табл. 5.5).

Таблица 5.5 Структура БД

Таблиця	Поле	Тип данних	Ключі	Обмеження
Object_types	object_type_id	INTEGER	PK	Не пустий
	name	VARCHAR2(20)		Не пустий
Objects	object_id	INTEGER	PK	Не пустий
	name	VARCHAR2(200)		Не пустий
	object_type_id	INTEGER	FK - Object_types	Не пустий
Attributes	attr_id	INTEGER	PK	Не пустий
	name	VARCHAR2(20)		Не пустий
	object_type_id	INTEGER	FK - Object_types	Не пустий
Params	text_value	VARCHAR2(200)		
	number_value	INTEGER		
	date_value	DATE		
	attr_id	INTEGER	FK - Attributes	Не пустий
	object_id	INTEGER	FK - Objects	Не пустий

5.3. Фізична реалізація БД метамоделі та міграція даних

Сценарій створення таблиць БД ІС у випадку метамоделі:

```
Create table "Object_types" (
     "object_type_id" Integer NOT NULL ,
     "name" Varchar2(20) NOT NULL ,
primary key ("object_type_id")
) ;

Create table "Objects" (
     "object_id" Integer NOT NULL ,
     "name" Varchar2(200) NOT NULL ,
     "object_type_id" Integer NOT NULL ,
primary key ("object_id")
) ;

Create table "Attributes" (
     "attr_id" Integer NOT NULL ,
     "name" Varchar2(20) NOT NULL ,
     "object_type_id" Integer NOT NULL ,
primary key ("attr_id")
) ;

Create table "Params" (
     "text_value" Varchar2(200),
     "number_value" Integer,
     "date_value" Date,
     "attr_id" Integer NOT NULL ,
     "object_id" Integer NOT NULL
) ;
```

Перенесемо дані з основної БД до новостворених таблиць.

```
Insert into ….
```

5.4. Реалізація запитів в разі застосування метамоделі

1. Надання відомостей про мешканців певної квартири (будинку).

```
select …
```

last_name	first_name	mid_name	sex	birthday	passport
Коломиец	Наталья	Петровна	1	27.09.02	МВ 885393
Коломиец	Татьяна	Николаевна	1	13.07.89	МВ 112835
Коломиец	Роман	Викторович	0	09.12.50	МВ 207375

Висновки

При виконанні курсової роботи була спроектована і реалізована база даних ІС МОЗ. Побудовані DFD і ERD діаграми концептуальної моделі даної інформаційної системи. Реалізовано сценарії створення і заповнення бази даних, а також реалізовані типові запити, а саме:

- Надання відомостей про мешканців певної квартири (будинку).
- Надання відомостей про певний житлі з урахуванням всіх його адрес.
- …

Для одного із запитів був проведений аналіз плану виконання запиту. Для поліпшення швидкодії …

Спроектована БД була відображена на метамодель по А.Тенцеру.

В ході виконання курсової роботи було помічено, що ефективність виконання запитів даної бази даних можна поліпшити, додавши індекси на стовпці, які часто використовуються в умовах запитів.

Список використаних джерел

1. А.Чекалов. Базы данных: от проектирования до разработки приложений - СПб.: БХВ-Петербург, 2003. — 384 с.
2. Анатолий Тенцер. База данных – хранилище объектов [Электронный ресурс] - М.: «КомпьютерПресс», 2001 — Режим доступу до журн.: www.compress.ru/article.aspx?id=11515
3. Слайды лекций дисциплины «Информационные системы и базы данных» [Электронный ресурс] — Режим доступу: http://dl.sumdu.edu.ua/e-pub/db/

Додаток А. Сценарій заповнення БД ІС

```
Insert into "hospital" values (1, 'Центральная городская
больница №1');
Insert into "hospital" values (4, 'Городская больница №4');
Insert into "hospital" values (5, 'Городская клиническая
больница №5');

Insert into "adds" values (1, 'Петропавловская','74',1);
Insert into "adds" values (2, 'Калинина','15',1);
Insert into "adds" values (3, 'Соборная','42',1);
Insert into "adds" values (4, 'Супруна','34',4);
Insert into "adds" values (5, 'Металлургов','7А',4);
Insert into "adds" values (6, 'Ахтырская','12',5);
Insert into "adds" values (7, 'Харьковская','96',5);
Insert into "adds" values (8, 'Парковая','4',5);
Insert into "adds" values (9, 'Харьковская','106',5);

Insert into "apts" values (3,1);
Insert into "apts" values (0,8);
Insert into "apts" values (0,2);
Insert into "apts" values (12,3);
Insert into "apts" values (25,4);
Insert into "apts" values (36,4);
Insert into "apts" values (3,5);
Insert into "apts" values (8,5);
Insert into "apts" values (13,6);
Insert into "apts" values (46,7);
Insert into "apts" values (12,9);
Insert into "apts" values (35,9);
Insert into "apts" values (28,9);
Insert into "people" values
...
```