Задания по запросам. Спортмастер

1. Создать и заполнить таблицы
2. Написать запросы, обосновав выбор их структуры
3. Если это возможно, привести несколько решений
4. Подтвердить правильность решений
5. Проанализировать планы исполнения

Варианты заданий

1. Схема

таблица product (продукт), поля: id, name (название), product\_id (айди продукта)

таблица order (порядок), поля: id, product\_id, category\_id (айди категории)

таблица category (категория), поля: id, name (название)

Запросы:

(1) достать все товары вместе с их категориями, (2) достать товар 'Огурец' вместе с его категориями, (3) достать все товары из категории 'Овощи', (4) достать все товары, которые принадлежат более чем одной категории.

1. Схема

таблица product (товар), поля: id, name (название), price (цена), date (дата), product\_id (айди продукта)

таблица purchase (покупка), поля: id, name (название), user\_id (айди пользователя), product\_id (айди продукта)

таблица user (пользователь), поля: id, name (имя)

Запросы:

1. вывести пользователей вместе с их покупками, (2) вывести пользователей вместе с суммами всех их покупок, (3) найти суммарные покупки на сайте за определенный месяц, (4) найти суммарные покупки на сайте помесячно (то есть результат будет в таком виде: март 2010 — сумма1, апрель 2010 — сумма2, май 2010 — сумма3 и тд).
2. Схема

таблица game (игра) поля: id, team1\_id (айди 1й команды), team2\_id (айдм 2й команды), date (дата), status (статус)

таблица team (команда) поля: id, name (название)

Запросы:

Есть сайт с датами футбольных игр. В каждой игре нужно выводить дату игры, первую команду и вторую команду. После того, как игра прошла — нужно выводить еще и счет. **Запросы:** (1) получить все игры вместе с командами, (2) получить все игры с командами за текущий месяц, (3) получить все игры с командами за предыдущий месяц, (4) получить все сыгранные игры, (5) получить все несыгранные игры.

1. Схема:

**таблица** user (пользователь), **поля**: id, name (имя), mother\_id (айди матери), father\_id (айди отца), sex (стать)

Запросы:

Генеологическое дерево. Пользователь, его бабушки, дедушки, мама, папа, братья, сестры, дети. Можно найти любого родственника в любом колене (например, пра-пра-пра-дедушку). То есть есть пользователь, он может быть сыном, отцом, братом, дедушкой, прадедушкой и все это одновременно. Нужно хранить и получать родственные связи. Запросы: (1) получить отца пользователя, (2) получить мать пользователя, (3) получить детей пользователя, (4) получить сыновей пользователя, (5) получить дочерей пользователя, (6) получить внуков пользователя, (7) получить бабушек пользователя, (8) получить дедушек пользователя, (9) получить пра-пра-пра-дедушку пользователя, (10) получить братьев и сестер пользователя, (11) получить братьев пользователя

1. Схема:

таблица goods (товар), поля: id, name (название), quantity(количество), price(цена), sub\_category\_id (айди подкатегории)

таблица sub\_category (подкатегория), поля: id, name (название), price (цена), category\_id (айди категории)

таблица category (категория), поля: id, name (название)

Товар (название, цена, количество), подкатегория товара, категория товара. Товар принадлежит подкатегории, подкатегория — категории. Пример: помидорки черри (товар), помидоры (подкатегория), овощи (категория).

**Запросы:**

1. достать товары вместе с подкатегориями и категориями, (2) достать товары из подкатегории 'Помидоры', (3) достать все подкатегории категории 'Овощи'.
2. Схема:

таблица river (река), поля: id, name (название), trid\_id (айди притока)

таблица tributaries (приток), поля: id, name (название), sea\_id (айди моря), river\_id (айди реки)

таблица sea (море), поля: id, name (название)

Море, реки, притоки (притоки делятся на правые и левые). Реки могут быть притоками других рек или впадать прямо в море.

 Запросы: (1) получить все реки Черного Моря, (2) получить все реки Черного Моря вместе с притоками, (3) получить все притоки реки Днепр, (4) получить куда впадает данная река (в какую реку или в какое море)

1. Схема:

таблица user (пользователь), поля: id, name (название), city\_id (айди города)

таблица city (город), поля: id, name (название)

Запросы:

1. достать пользователей вместе с их городом, (2) достать все города, (3) достать всех пользователей из города Минск, (4) достать все города, в которых есть пользователи, (5) достать все города, в которых нет пользователей, (6) вывести список городов с количеством пользователей в них, (7) вывести список городов, в которых количество пользователей больше трех.
2. Схема:

таблица user (пользователь), поля: id, name (имя), surname (фамилия), age (возраст), address (адрес), family\_id (айди семьи)

Есть мужья и жены, а также неженатые/незамужние. Для всех указывается имя, фамилия, возраст, адрес. Муж с женой имеют одну фамилию и живут по одному адресу.

Запросы:

(1) получить мужей с женами и наоборот, (2) получить холостых, (3) получить семьи (муж+жена), семьи не должны дублироваться (пара должна быть только 1 раз)

1. Схема:

таблица user (пользователь), поля: id, name (имя), user\_id (айди человека)

таблица order (порядок), поля: id, user\_id (айди человека), son\_id (айди сина)

таблица son (интерес), поля: id, name (название)

Есть отцы и сыновья. У отца может быть много сыновей.

Запросы:

1. получить всех сыновей пользователя, (2) получить отца пользователя, (3) получить дедушку пользователя, (4) получить внуков пользователя
2. Сложное задание:

Есть работники с разными профессиями. У каждой профессии фиксированная зарплата за 1 рабочий день, эта зарплата каждый месяц меняется (из-за инфляции). Каждый работник работает в месяц разное количество дней. Получите зарплаты всех работников за предыдущий месяц. Иногда может понадобится получить зарплаты за какой-то месяц какого-либо года или все зарплаты работника за год (напишите соответствующие запросы). Есть ли смысл каждый раз высчитывать это или лучше рассчитанные и выданные зарплаты как-то хранить? Если хранить, то как это реализовать?

1. Сложное задание:

Программы для генерации тестовых данных.

Пример разбора задания

Для каждого менеджера выведите его порядковый номер, имя, и наиболее высокооплачиваемого сотрудника этого менеджера. Сотрудники, чей менеджер не определен, в запросе не учитываются.

SELECT e1.employee id, e1.first name, e2.first name, e2.salary

FROM

employees e1 JOIN employees e2 ON(e1.employee\_id = e2.manager\_id)

WHERE

e2.salary =

(SELECT max(e3.salary) FROM employees e3 WHERE e3.manager\_id = e1.employee\_id);

Создайте запрос, который возвращает имя, должность и зарплату для всех сотрудников, чья зарплат выше средней, и чьим коллего ??

SELECT e.first\_name, j.job\_title, e.salary FROM

employees e JOIN

jobs j ON(e.job\_id = j.job\_id) JOIN

employees e1 ON(e1.department\_id = e.department\_id)

WHERE

e.salary > (SELECT AVG(e2.salary) FROM employees e2) AND

e1.first\_name LIKE '%T%' AND

e1.first\_name NOT LIKE e.first\_name;