Автосалон «Врум-Бум» 27 09 2024 Демидова Татьяна

Шаг 1. Скачайте и проанализируйте исходный CSV-файл с данными.
Скачать файл.
Файл скачан. Данные представлены в виде 8 столбцов и 1000 строк (без учета строки с назвваниями столбцов).

Шаг 2. Начиная с этого пункта (включительно) все SQL-команды должны быть отправлены на проверку.
Создайте схему raw_data и таблицу sales в этой схеме.

-- Создание схемы raw_data

```
CREATE SCHEMA IF NOT EXISTS raw_data;
```

-- Создание таблицы sales

```
CREATE TABLE IF NOT EXISTS raw_data.sales (
    id SERIAL PRIMARY KEY,
    auto VARCHAR(100),
    gasoline_consumption NUMERIC(3, 1) NULL,
    price NUMERIC(9, 2) NOT NULL,
    date DATE NOT NULL,
    person VARCHAR(100),
    phone VARCHAR(30),
    discount NUMERIC(4, 0),
    brand_origin VARCHAR(100)
);
```

Шаг 3. Заполните таблицу sales данными, используя команду COPY в менеджере БД (например, DBeaver) или \copy в psql. Если возникнет ошибка о лишних данных в исходном файле, укажите явно параметр DELIMITER.

-- Заполнение таблицы sales данными в DBeaver

```
INSERT INTO raw_data.sales (id, auto, gasoline_consumption, price, date, person, phone, discount, brand_origin)
VALUES
(1, 'Lada Vesta, grey', 7.3, 11243.44, '2019-12-15', 'Michael Wu', '+1-587-114-0889x3149', 20, 'Russia'),
(2, 'BMW F80, red', 8.3, 63761.75, '2019-01-29', 'Carla Smith', '001-818-501-7528x0438', 0, 'Germany');
```

… и так далее до 1000 строк.

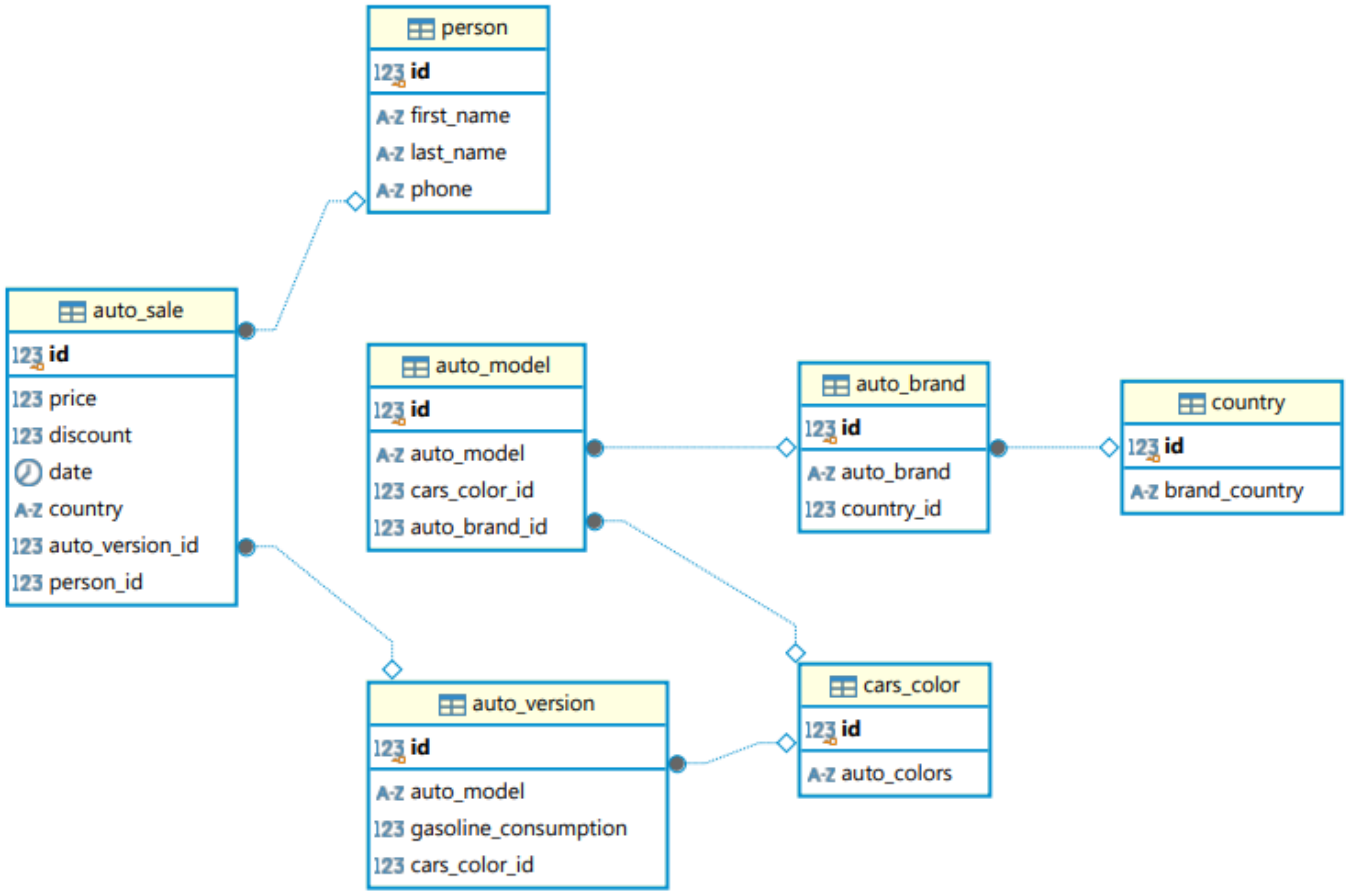
-- Заполнение таблицы sales данными в psql

```
\copy raw_data.sales (id, auto, gasoline_consumption, price, sale_date, person, phone, discount, brand_origin) FROM 'C:\Temp\cars.csv' WITH (FORMAT csv, HEADER, DELIMITER ',', NULL 'null');
```

Шаг 4. Проанализируйте сырые данные. Подумайте:

- какие данные повторяются и их стоит вынести в отдельные таблицы;
- какие типы данных должны быть у каждого поля;
- на какие поля стоит добавить ограничения (CONSTRAINTS);
- какие поля могут содержать NULL, а где нужно добавить ограничение NOT NULL.

Обратите внимание на атрибут color. Подумайте, как лучше его хранить. У вас будет связь многие-ко-многим, ведь у одного цвета может быть несколько машин, а машина может быть разных цветов.



Шаг 5. Создайте схему car_shop, а в ней создайте нормализованные таблицы (до третьей нормальной формы). Подумайте, какие поля будут выступать первичными ключами, а какие внешними**.** У всех первичных ключей должен быть автоинкремент. В отдельном файле опишите, почему вы выбрали такую модель данных.

-- Создание схемы car_shop

CREATE SCHEMA IF NOT EXISTS car_shop;

Шаг 6. Выберите наиболее подходящий формат данных для каждой колонки. Используя комментарий (/*comment*/) для каждого поля, напишите, почему вы выбрали тот или иной тип данных. Например:

- brand_name varchar — в названии бренда могут быть и цифры, и буквы, поэтому выбираем varchar.
- price numeric(9, 2) — цена может содержать только сотые и не может быть больше семизначной суммы. У numeric повышенная точность при работе с дробными числами, поэтому при операциях с этим типом данных, дробные числа не потеряются.

```
CREATE SCHEMA IF NOT EXISTS car_shop;

ALTER TABLE car_shop.auto_sale
DROP COLUMN cars_id;

DROP TABLE car_shop.auto_sale;
DROP SCHEMA car_shop CASCADE;

CREATE TABLE IF NOT EXISTS car_shop.country (
    id SERIAL PRIMARY KEY,
    brand_country VARCHAR(100)
);
```

```
CREATE TABLE IF NOT EXISTS car_shop.auto_brand (
    id SERIAL PRIMARY KEY,
    auto_brand VARCHAR(50),
    country_id INT REFERENCES car_shop.country(id)
);


CREATE TABLE IF NOT EXISTS car_shop.cars_color (
    id SERIAL PRIMARY KEY,
    auto_colors VARCHAR(60)
);


CREATE TABLE IF NOT EXISTS car_shop.auto_model (
    id SERIAL PRIMARY KEY,
    auto_model VARCHAR(50),
    cars_color_id INT REFERENCES car_shop.cars_color(id),
    auto_brand_id INT REFERENCES car_shop.auto_brand(id)
);


CREATE TABLE IF NOT EXISTS car_shop.person (
    id SERIAL PRIMARY KEY,
    first_name VARCHAR(100),
    last_name VARCHAR(100),
    phone VARCHAR(30)
);


CREATE TABLE IF NOT EXISTS car_shop.auto_version (
    id SERIAL PRIMARY KEY,
    auto_model VARCHAR(50),
    gasoline_consumption NUMERIC(3, 1) NULL,
    cars_color_id INT REFERENCES car_shop.cars_color(id)
);


CREATE TABLE IF NOT EXISTS car_shop.auto_sale (
    id SERIAL PRIMARY KEY,
    price NUMERIC(9, 2) NOT NULL,
    discount NUMERIC(4, 0),
    date DATE NOT NULL,
    country VARCHAR(50),
    auto_version_id INT REFERENCES car_shop.auto_version(id),
    person_id INT REFERENCES car_shop.person(id),
    UNIQUE (price, discount, date, country)
);
```

Шаг 7. Заполните все таблицы данными, с помощью команд INSERT INTO … SELECT …. Чтобы разбить столбец с маркой, моделью машины и её цветом, можно использовать разные способы — выбирайте любой, удобный для себя.

```
INSERT INTO car_shop.country (brand_country)
SELECT DISTINCT COALESCE(brand_origin, 'Not Info')
FROM raw_data.sales
ORDER BY COALESCE(brand_origin, 'Not Info');
```

```
INSERT INTO car_shop.auto_brand (auto_brand, country_id)
SELECT DISTINCT
   SPLIT_PART(rds.auto, ' ', 1) AS brand,
   csc.id AS country_id
FROM raw_data.sales rds
JOIN car_shop.country csc
   ON COALESCE(rds.brand_origin, 'Not Info') = csc.brand_country
ORDER BY brand;


INSERT INTO car_shop.cars_color (auto_colors)
SELECT DISTINCT
   TRIM(SPLIT_PART(auto, ',', 2)) AS auto_colors
FROM raw_data.sales
ORDER BY auto_colors;


INSERT INTO car_shop.auto_model (auto_model, cars_color_id, auto_brand_id)
SELECT
   TRIM(SPLIT_PART(SPLIT_PART(rds.auto, ',', 1), ' ', 2)) ||
   CASE WHEN LENGTH(SPLIT_PART(SPLIT_PART(rds.auto, ',', 1), ' ', 3)) > 0
       THEN ' ' || SPLIT_PART(SPLIT_PART(rds.auto, ',', 1), ' ', 3)
       ELSE ''
   END AS auto_model,
   cscc.id AS cars_color_id,
   csab.id AS auto_brand_id
FROM raw_data.sales rds
JOIN car_shop.auto_brand csab
   ON SPLIT_PART(rds.auto, ' ', 1) = csab.auto_brand
JOIN car_shop.cars_color cscc
   ON TRIM(SPLIT_PART(rds.auto, ',', 2)) = cscc.auto_colors
GROUP BY auto_model, cars_color_id, auto_brand_id
ORDER BY auto_model;


INSERT INTO car_shop.person (first_name, last_name, phone)
SELECT DISTINCT ON (TRIM(BOTH ' ' FROM SPLIT_PART(rds.person, ' ', 1)),
                   INITCAP(TRIM(BOTH ' ' FROM SPLIT_PART(rds.person, ' ', 2))),
                   rds.phone)
   TRIM(BOTH ' ' FROM SPLIT_PART(rds.person, ' ', 1)) AS first_name,
   INITCAP(TRIM(BOTH ' ' FROM SPLIT_PART(rds.person, ' ', 2))) AS last_name,
   rds.phone
FROM raw_data.sales rds
JOIN car_shop.auto_brand csab
   ON SPLIT_PART(rds.auto, ' ', 1) = csab.auto_brand
ORDER BY first_name, last_name, phone;


INSERT INTO car_shop.auto_version (auto_model, gasoline_consumption, cars_color_id)
SELECT DISTINCT
   csam.auto_model AS auto_model,
   rds.gasoline_consumption,
   cscc.id
FROM raw_data.sales rds
JOIN car_shop.auto_model csam ON TRIM(SPLIT_PART(SPLIT_PART(rds.auto, ',', 1), ' ', 2)) ||
   CASE WHEN LENGTH(TRIM(SPLIT_PART(SPLIT_PART(rds.auto, ',', 1), ' ', 3))) > 0
       THEN ' ' || TRIM(SPLIT_PART(SPLIT_PART(rds.auto, ',', 1), ' ', 3))
       ELSE ''
   END = csam.auto_model
JOIN car_shop.cars_color cscc ON TRIM(BOTH ' ' FROM SPLIT_PART(rds.auto, ',', 2)) = cscc.auto_colors
```

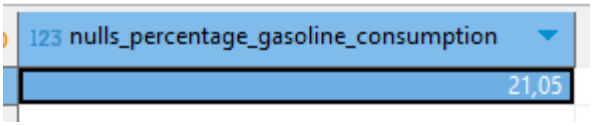
```
ORDER BY csam.auto_model;


INSERT INTO car_shop.auto_sale (price, discount, date, country, auto_version_id, person_id)
SELECT DISTINCT
    rds.price,
    rds.discount,
    rds.date,
    rds.brand_origin,
    csav.id AS auto_version_id,
    csp.id AS person_id
FROM raw_data.sales rds
RIGHT JOIN car_shop.auto_version csav
    ON TRIM(SPLIT_PART(SPLIT_PART(rds.auto, ',', 1), ' ', 2)) ||
    CASE WHEN LENGTH(TRIM(SPLIT_PART(SPLIT_PART(rds.auto, ',', 1), ' ', 3))) > 0
        THEN ' ' || TRIM(SPLIT_PART(SPLIT_PART(rds.auto, ',', 1), ' ', 3))
        ELSE ''
    END = csav.auto_model
RIGHT JOIN car_shop.person csp
    ON rds.phone = csp.phone
ON CONFLICT (price, discount, date, country)
DO NOTHING;
```

Задание 1 из 6
Напишите запрос, который выведет процент моделей машин, у которых нет параметра gasoline_consumption.

```
SELECT
    ROUND((COUNT(id) FILTER (WHERE gasoline_consumption IS NULL) * 100.0)
    /
    COUNT(id), 2) AS nulls_percentage_gasoline_consumption
FROM car_shop.auto_version;
```

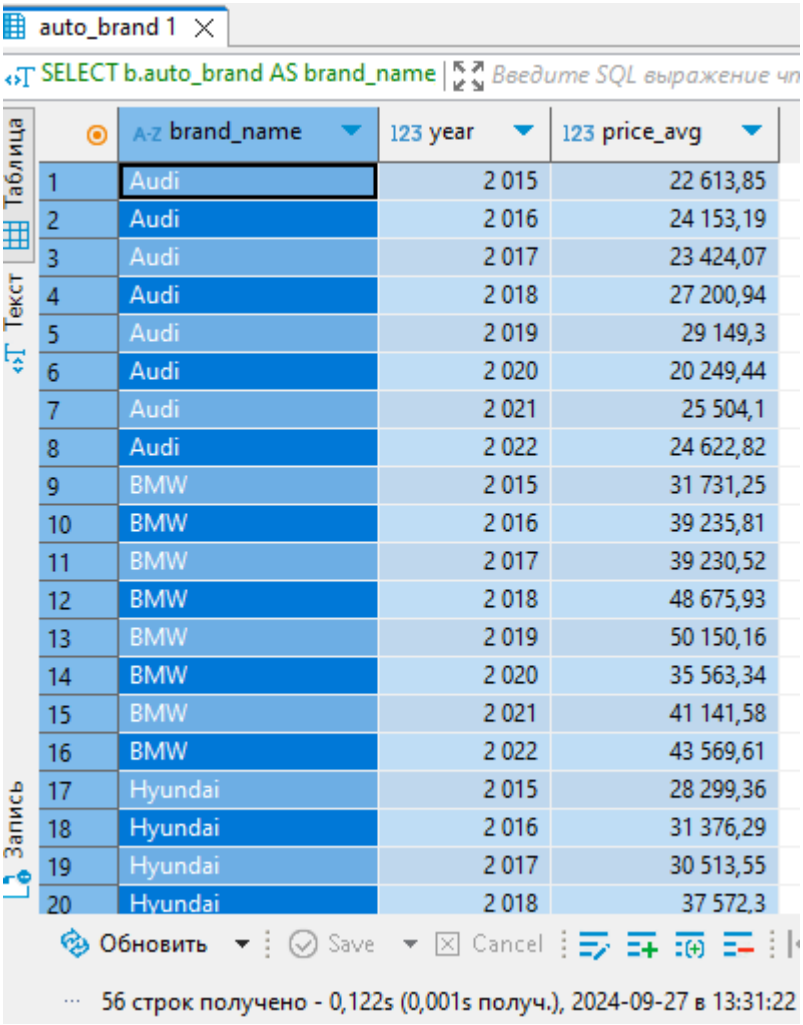
Вот формат итоговой таблицы:



Задание 2 из 6
Напишите запрос, который покажет название бренда и среднюю цену его автомобилей в разбивке по всем годам с учётом скидки. Итоговый результат отсортируйте по названию бренда и году в восходящем порядке. Среднюю цену округлите до второго знака после запятой.

```
SELECT
    b.auto_brand AS brand_name,
    EXTRACT(YEAR FROM s.date) AS year,
    ROUND(AVG(s.price * (1 - s.discount / 100)), 2) AS price_avg
FROM car_shop.auto_sale AS s
LEFT JOIN car_shop.auto_version AS csav ON s.auto_version_id = csav.id
LEFT JOIN car_shop.auto_model AS m ON csav.auto_model = m.auto_model
LEFT JOIN car_shop.auto_brand AS b ON m.auto_brand_id = b.id
GROUP BY b.auto_brand, year
ORDER BY b.auto_brand ASC, year ASC;
```

Вот формат итоговой таблицы:



	brand_name	year	price_avg
1	Audi	2 015	22 613,85
2	Audi	2 016	24 153,19
3	Audi	2 017	23 424,07
4	Audi	2 018	27 200,94
5	Audi	2 019	29 149,3
6	Audi	2 020	20 249,44
7	Audi	2 021	25 504,1
8	Audi	2 022	24 622,82
9	BMW	2 015	31 731,25
10	BMW	2 016	39 235,81
11	BMW	2 017	39 230,52
12	BMW	2 018	48 675,93
13	BMW	2 019	50 150,16
14	BMW	2 020	35 563,34
15	BMW	2 021	41 141,58
16	BMW	2 022	43 569,61
17	Hyundai	2 015	28 299,36
18	Hyundai	2 016	31 376,29
19	Hyundai	2 017	30 513,55
20	Hyundai	2 018	37 572,3

Задание 3 из 6
Посчитайте среднюю цену всех автомобилей с разбивкой по месяцам в 2022 году с учётом скидки. Результат отсортируйте по месяцам в восходящем порядке. Среднюю цену округлите до второго знака после запятой.

```
SELECT
    EXTRACT(MONTH FROM date) AS month,
    ROUND(AVG(price * (1 - discount / 100)), 2) AS price_avg
FROM car_shop.auto_sale
WHERE EXTRACT(YEAR FROM date) = 2022
GROUP BY month
ORDER BY month ASC;
```

Вот формат итоговой таблицы:

	month	price_avg
1	1	37 388,15
2	2	35 186,4
3	3	45 521,52
4	4	31 545,32
5	5	31 683,01
6	6	28 731,78
7	7	30 202,76
8	8	37 881,63
9	9	29 725,9
10	10	40 822,78
11	11	22 650,24
12	12	32 102,63

Задание 4 из 6

Используя функцию STRING_AGG, напишите запрос, который выведет список купленных машин у каждого пользователя через запятую. Пользователь может купить две одинаковые машины — это нормально. Название машины покажите полное, с названием бренда — например: Tesla Model 3. Отсортируйте по имени пользователя в восходящем порядке. Сортировка внутри самой строки с машинами не нужна.

```
SELECT
    p.first_name || ' ' || p.last_name AS person,
    STRING_AGG(TRIM(SPLIT_PART(s.auto, ',', 1)), ', ') AS cars
FROM raw_data.sales AS s
JOIN car_shop.person AS p USING(phone)
GROUP BY p.first_name, p.last_name
ORDER BY person ASC;
```

Вот формат итоговой таблицы:

	person	cars
881	William Chambers	Lada Vesta
882	William Fleming	Tesla Model 3
883	William Hess	BMW F80
884	William Hunt	Kia Rio
885	William Kirk	Hyundai Elantra
886	William Thompson	BMW F30
887	William Wong	Audi S3
888	Yolanda Wilson	Audi S3
889	Zachary Buckley	Audi RS3
890	Zachary Long	Audi A3
891	Zachary Montes	BMW F80, Tesla Model X
892	Zachary Robinson	Kia Rio

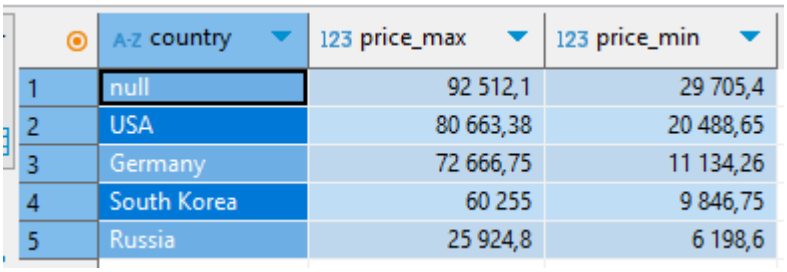
Задание 5 из 6

Напишите запрос, который вернёт самую большую и самую маленькую цену продажи автомобиля с разбивкой по стране без учёта скидки. Цена в колонке price дана с учётом скидки.

```
SELECT
    country AS country,
    ROUND(MAX((csas.price * 100) / (100 - csas.discount)), 2) AS price_max,
    ROUND(MIN((csas.price * 100) / (100 - csas.discount)), 2) AS price_min
FROM car_shop.auto_sale AS csas
```

GROUP BY *country*
ORDER BY *price_max* **DESC**;

Вот формат итоговой таблицы:



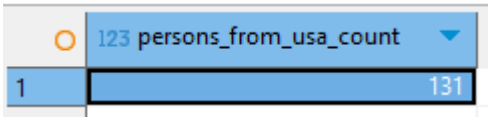
	country	price_max	price_min
1	null	92 512,1	29 705,4
2	USA	80 663,38	20 488,65
3	Germany	72 666,75	11 134,26
4	South Korea	60 255	9 846,75
5	Russia	25 924,8	6 198,6

Задание 6 из 6

Напишите запрос, который покажет количество всех пользователей из США. Это пользователи, у которых номер телефона начинается на +1.

SELECT COUNT(*) **AS** *persons_from_usa_cou*nt
FROM car_shop.pers**on**
WHERE phone **LIKE** '+1%';

Вот формат итоговой таблицы:
Как оформить решение



	persons_from_usa_count
1	131

27.09.2024