Для решения задач использовался PostgreSQL.

Задача 1

Задание: необходимо найти 3-х самых молодых сотрудников в коллективе и выдать их имена, предварительно отсортировав. Задачу требуется решить несколькими способами (чем больше, тем лучше).

Можно использовать следующий ddl:

```
create table test (id number, name varchar2(10), age number);
insert into test values (1, 'Вася', 23);
insert into test values (2, 'Петя', 40);
insert into test values (3, 'Маша', 19);
insert into test values (4, 'Марина', 23);
insert into test values (5, 'Сергей', 34);
```

Решение:

```
SELECT name FROM test
ORDER BY age
LIMIT 3;
```

Для конкретно данной таблицы можно использовать также следующие варианты:

```
SELECT name FROM test
WHERE age < (
                    SELECT AVG(AGE)
                    FROM TEST
);
```

```
SELECT name FROM test
WHERE age BETWEEN (
                    SELECT MIN(age) FROM test)
                    AND (
                    SELECT MIN(age) FROM test
                    WHERE age NOT IN (
                        SELECT MIN(age) FROM test));
```

Задача 2

Есть таблица:

abonent	region_id	dttm
7072110988	32722	2021-08-18 13:15
7072110988	32722	2021-08-18 14:00
7072110988	21534	2021-08-18 14:15
7072110988	32722	2021-08-19 09:00
7071107101	12533	2021-08-19 09:15
7071107101	32722	2021-08-19 09:27

Описание атрибутов:

1) abonent – номер абонента;
2) region_id – id региона в котором находится абонент;
3) dttm – день и время звонка.

Задание: нужно для каждого дня определить последнее местоположение абонента.

То есть нужно вывести:

abonent	region_id	dttm
7072110988	21534	2021-08-18 14:15
7072110988	32722	2021-08-19 09:00
7071107101	32722	2021-08-19 09:27

Решение:

```
WITH uniq_date AS
(
SELECT * FROM (
   SELECT abonent, region_id, dttm, row_number() OVER
      (PARTITION BY abonent, date(dttm)
       ORDER BY dttm DESC) AS row_num
   FROM clients) AS subquery
)
SELECT abonent, region_id, dttm
FROM uniq_date
WHERE row_num = 1
ORDER BY abonent DESC;
```

Для данной таблицы можно использовать также следующий вариант:

```
SELECT abonent, region_id, MAX(dttm)
FROM clients
GROUP BY abonent, region_id
ORDER BY abonent DESC
LIMIT 3;
```

Задача 3

Есть таблица item_prices:

Название столбца	Тип	Описание
item_id	number(21,0)	Идентификатор товара
item_name	varchar2(150)	Название товара
item_price	number(12,2)	Цена товара
created_dttm	timestamp	Дата добавления записи

Задание: необходимо сформировать таблицу следующего вида dict_item_prices.

Название столбца	Тип	Описание
item_id	number(21,0)	Идентификатор товара
item_name	varchar2(150)	Название товара
item_price	number(12,2)	Цена товара
valid_from_dt	date	Дата, с которой начала действовать данная цена (created_dttm записи с ценой)
valid_to_dt	date	Дата, до которой действовала данная цена (created_dttm следующей записи по данному товару «минус» один день)

Примечание: для последней (действующей на данный момент) цены устанавливается дата 9999-12-31.

Решение:

```
CREATE TABLE dict_item_prices AS
SELECT item_id, item_name, item_price, created_dttm::DATE AS valid_from_dt,
                        COALESCE(LEAD(created_dttm::DATE - 1) OVER
                                (PARTITION BY item_id
                                 ORDER BY item_id, created_dttm::DATE), '9999-12-31') AS valid_to_dt
FROM item_prices;
```

Задача 4

Есть исходная таблица детализации по транзакциям transaction_details:

Название столбца	Тип	Описание
transaction_id	number(21,0)	Идентификатор транзакции
customer_id	number(21,0)	Идентификатор клиента
item_id	number(21,0)	Идентификатор товара
item_number	number(8,0)	Количество купленных единиц товара
transaction_dttm	timestamp	Дата-время транзакции

Задание: необходимо сформировать таблицу следующего вида customer_aggr:

Название столбца	Тип	Описание
customer_id	number(21,0)	Идентификатор клиента
amount_spent_1m	number(12,2)	Потраченная клиентом сумма за последний месяц
top_item_1m	varchar2(150)	Товар (item_name), на который за последний месяц клиент потратил больше всего

Примечание:

- при расчете необходимо учитывать актуальную цену на момент совершения транзакции (по справочнику dict_item_prices из Задача 3);
- клиенты, не совершавшие покупок в последний месяц, в итоговую таблицу не попадают;
- последний месяц определяется как последние 30 дней на момент построения отчета.

Решение:

```
WITH top_item AS
(
SELECT customer_id, SUM(CAST (item_number AS Decimal)  * dip.item_price) AS amount_spent_1m, dip.item_name
FROM transaction_details
                    INNER JOIN dict_item_prices AS dip USING(item_id)
WHERE transaction_dttm  > NOW() - INTERVAL '30 DAY'
                    AND transaction_dttm BETWEEN dip.valid_from_dt
                    AND dip.valid_to_dt
GROUP BY customer_id, dip.item_name
ORDER BY amount_spent_1m  DESC, customer_id
LIMIT 3
)

SELECT customer_id, SUM(CAST (item_number AS Decimal) * dip.item_price) AS amount_spent_1m,
MAX(top_item.item_name) AS top_item_1m
FROM transaction_details
                    INNER JOIN dict_item_prices AS dip USING(item_id)
                    INNER JOIN top_item USING(customer_id)
WHERE transaction_dttm  > NOW() - INTERVAL '30 DAY'
                    AND transaction_dttm BETWEEN dip.valid_from_dt AND dip.valid_to_dt
GROUP BY customer_id ;
```

Задача 5

Существует таблица публикаций posts в социальных сетях с указанием даты и названием публикации.

Задание: рассчитать количество публикаций в месяц с указанием первой даты месяца и долей увеличения количества сообщений (публикаций) относительно предыдущего месяца.

Данные в результирующей таблице должны быть упорядочены в хронологическом порядке.

Примечание: доля увеличения количества сообщений может быть отрицательной, а результат должен быть округлен до одного знака после запятой с добавлением знака %.

Table posts (пример)

id	created_at	title
1	2022-01-17 08:50:58	Sberbank is the best bank
2	2022-01-17 18:36:41	Visa vs Mastercard
3	2022-01-17 16:16:17	Visa vs UnionPay
4	2022-01-17 18:01:00	Mastercard vs UnionPay
5	2022-01-16 16:44:36	Hadoop or Greenplum: pros and cons
6	2022-01-16 14:57:32	NFC: wireless payment

Table results (пример для наглядного результата, с таблицей posts не соотносится 1 к 1)

dt	count	prcnt_growth
2022-02-01	175	null
2022-03-01	338	93.1%
2022-04-01	345	2.1%
2022-05-01	295	-14.5%
2022-06-01	330	11.9%

Решение:

```
WITH row_pbm AS
(
SELECT dt, count, row_number() OVER
                          (ORDER BY dt) AS row_num1
   FROM (
                          SELECT CAST(date_trunc('month', created_at) AS DATE) AS dt, COUNT(title)
                          FROM posts
                          GROUP BY dt
                          ORDER BY dt) AS pbm
)

SELECT t1.dt, t1.count, ROUND((t2.count - t1.count) * 1.0 / t1.count * 100,1) AS prcnt_growth
FROM row_pbm AS t1
LEFT JOIN row_pbm AS t2 on t1.row_num1 = t2.row_num1 + 1;
```

Улучшить данный запрос можно заменив null на 0 а так же добавить защиту деления на ноль:

```
SELECT t1.dt, t1.count, ROUND(COALESCE((t2.count - t1.count) * 1.0 / NULLIF (t1.count, 0),0) * 100,1) AS
prcnt_growth
```