**SQL1. Базовые запросы\_ДЗ 1.1. Часть 1**

SELECT \*

FROM netflix

WHERE type like 'Movie'

and cast\_names like '%Joaquin Phoenix%'

**SQL1. Базовые запросы\_ДЗ 1.1. Часть 2**

SELECT \*

FROM netflix

WHERE cast\_names like '%Renée Zellweger%'

and type like 'TV Show'

**SQL1. Базовые запросы\_ДЗ 1.1. Часть 3**

SELECT \*

FROM netflix

WHERE type like 'Movie'

and director in ('Sam Mendes', 'Shirish Kunder' , 'Quentin Tarantino', 'Martin Scorsese', 'Bong Joon Ho')

and release\_year >= '2010'

LIMIT 100

**SQL1. Базовые запросы\_ДЗ 1.1. Часть 4**

SELECT \*

FROM netflix

WHERE title in ('Joker', 'Judy', 'Once Upon a Time in... Hollywood', 'Marriage Story')

and (cast\_names like '%Joaquin Phoenix%'

or cast\_names like '%Renée Zellweger%'

or cast\_names like '%Brad Pitt%'

or cast\_names like '%Laura Dern%')

**SQL2. Генерация новых признаков и очистка данных\_ДЗ 2.1. Часть1**

SELECT game\_rank, name, genre, publisher, eu\_sales, platform\_name, sales\_start

, case when current\_date - sales\_start >= interval '9 years' then '40%'

else '0%'

end as discount\_rate

FROM game\_db

WHERE genre = 'Sports'

and platform\_name like '%PlayStation%'

and eu\_sales <= '0.2'

ORDER BY game\_rank asc

**SQL2. Генерация новых признаков и очистка данных\_ДЗ 2.1. Часть 2**

SELECT game\_rank, name, genre, publisher, eu\_sales, platform\_name, sales\_start

, case when current\_date - sales\_start >= interval '9 years' then '40%'

else '0%'

end as discount\_rate

, case when (lower(name) like '%fifa%'

or lower(name) like '%soccer%'

or lower(name) like '%football%') then 1

else 0

end as is\_soccer

FROM game\_db

WHERE genre = 'Sports'

and platform\_name like '%PlayStation%'

and eu\_sales <= '0.2'

ORDER BY game\_rank asc

**SQL3. Агрегатные функции\_ДЗ 3.1 Часть 1**

select platform\_name

, sum (global\_sales) total\_sales

, count (name) games

, count (publisher) publishers

, sum (global\_sales) / count (name) global\_sales\_per\_game

, sum (global\_sales) / count (publisher) sales\_per\_publisher

from game\_db

where sales\_start >= '2010-01-01'

and genre = 'Shooter'

group by platform\_name

having sum (global\_sales) >= 50

and count (name) >= 10

order by total\_sales desc

**SQL3. Агрегатные функции\_ДЗ 3.1 Часть 2**

select

total\_games, action\_shooter, adventure\_rpg, sport, others,

action\_shooter/total\_games::float as part\_action\_shooter,

adventure\_rpg/total\_games::float as part\_adventure\_rpg,

sport/total\_games::float as part\_sport,

strategy\_simulation/total\_games::float as part\_strategy\_simulation,

others/total\_games::float as part\_others

from

(

select

count (name) total\_games,

sum (case when genre in ('Action', 'Shooter') then 1 else 0 end) as action\_shooter,

sum (case when genre in ('Adventure', 'Role-Playing') then 1 else 0 end) as adventure\_rpg,

sum (case when genre in ('Fighting', 'Racing', 'Sports') then 1 else 0 end) as sport,

sum (case when genre in ('Strategy', 'Simulation') then 1 else 0 end) as strategy\_simulation,

sum (case when genre in ('Puzzle', 'Misc', 'Platform') then 1 else 0 end) as others

from game\_db

) as t

**SQL4. Объединение таблиц\_ДЗ 4. Задача 1**

select b.tconst, b."primaryTitle", r."averageRating"

from imdb.title\_basics b

left join imdb.title\_ratings r on b.tconst = r.tconst

where r."numVotes" >= 100000

and b."startYear" = 2020

order by r."averageRating" desc

limit 10

**SQL4. Объединение таблиц\_ДЗ 4. Задача 2**

select n."primaryName", avg(r."averageRating") as avg\_rating\_director

from imdb.title\_basics b

join imdb.title\_ratings r on b.tconst = r.tconst

join imdb.title\_crew\_long cl on b.tconst = cl.tconst

join imdb.name\_basics n on n.nconst = cl.directors

where r."numVotes" >= 100000

and b."titleType" in ('movie','tvSeries')

and b."startYear" = 2020

group by cl.directors, n."primaryName"

order by avg(r."averageRating") desc

limit 10

**SQL4. Объединение таблиц\_ДЗ 4. Задача 3**

select count(\*), "startYear"

from imdb.title\_basics b

where "startYear" >= 2015

and "startYear" <= 2020

group by "startYear"

order by "startYear" desc

**SQL4. Объединение таблиц\_ДЗ 4. Задача 4**

select '2018/2019' as title --, count(d18.directors), count(d19.directors)

, count(d19.directors)::float/count(d18.directors) part

from imdb.directors\_2018 d18

left join imdb.directors\_2019 d19 on d18.directors = d19.directors

union all

select '2019/2020' as title --, count(d19.directors), count(d20.directors)

, count(d20.directors)::float/count(d19.directors) part

from imdb.directors\_2019 d19

left join imdb.directors\_2020 d20 on d19.directors = d20.directors

**SQL5. Объединение таблиц. Часть2\_ДЗ 5. Задача 1**

SELECT 'Dallas' AS City,

avg(price)

FROM airbnb\_dallas.listings ld

WHERE ld.minimum\_nights >= 30

AND ld.room\_type = 'Private room'

UNION ALL

SELECT 'Oakland' AS City,

avg(price)

FROM airbnb\_oakland.listings lo

WHERE lo.minimum\_nights >= 30

AND lo.room\_type = 'Private room'

UNION ALL

SELECT 'New York' AS City,

avg(price)

FROM airbnb\_new\_york.listings ln

WHERE ln.minimum\_nights >= 30

AND ln.room\_type = 'Private room'

**SQL5. Объединение таблиц. Часть2\_ДЗ 5. Задача 2**

select 'Dallas' as city,

avg(case when c.date between '2021-03-01' and '2021-03-31' then replace(replace(c.adjusted\_price,'$',''),',','')::float else null end) as march,

avg(case when c.date between '2021-04-01' and '2021-04-30' then replace(replace(c.adjusted\_price,'$',''),',','')::float else null end) as april,

avg(case when c.date between '2021-05-01' and '2021-05-31' then replace(replace(c.adjusted\_price,'$',''),',','')::float else null end) as may

from airbnb\_dallas.listings l

join airbnb\_dallas.calendar c on l.id=c.listing\_id

where l.room\_type = 'Entire home/apt'

UNION ALL

select 'Oakland' as city,

avg(case when c.date between '2021-03-01' and '2021-03-31' then replace(replace(c.adjusted\_price,'$',''),',','')::float else null end) as march,

avg(case when c.date between '2021-04-01' and '2021-04-30' then replace(replace(c.adjusted\_price,'$',''),',','')::float else null end) as april,

avg(case when c.date between '2021-05-01' and '2021-05-31' then replace(replace(c.adjusted\_price,'$',''),',','')::float else null end) as may

from airbnb\_oakland.listings l

join airbnb\_oakland.calendar c on l.id=c.listing\_id

where l.room\_type = 'Entire home/apt'

UNION ALL

select 'New York' as city,

avg(case when c.date between '2021-03-01' and '2021-03-31' then replace(replace(c.adjusted\_price,'$',''),',','')::float else null end) as march,

avg(case when c.date between '2021-04-01' and '2021-04-30' then replace(replace(c.adjusted\_price,'$',''),',','')::float else null end) as april,

avg(case when c.date between '2021-05-01' and '2021-05-31' then replace(replace(c.adjusted\_price,'$',''),',','')::float else null end) as may

from airbnb\_new\_york.listings l

join airbnb\_new\_york.calendar c on l.id=c.listing\_id

where l.room\_type = 'Entire home/apt'

**SQL5. Объединение таблиц. Часть2\_ДЗ 5. Задача 3**

SELECT host\_id,

host\_name,

max(price)

FROM airbnb\_new\_york.listings l

WHERE l.host\_id in

(SELECT l.host\_id

FROM airbnb\_new\_york.listings l

JOIN airbnb\_new\_york.reviews r ON l.id = r.listing\_id

JOIN airbnb\_new\_york.neighbourhoods n ON l.neighbourhood = n.neighbourhood

WHERE r.date = '2020-04-01'

AND neighbourhood\_group = 'Brooklyn' )

GROUP BY l.host\_name,

l.host\_id

ORDER BY max(price) DESC

--LIMIT 1

**SQL6. Подзапросы и WITH\_ДЗ 6. Задача 1**

WITH top\_teacher AS

(SELECT count(id\_class),

id\_teacher

FROM skyeng\_db.classes

WHERE class\_status in ('success',

'failed\_by\_student')

AND date\_trunc('year', class\_start\_datetime) = '2016-01-01'

GROUP BY id\_teacher

ORDER BY 1 DESC

LIMIT 1)

SELECT \*

FROM skyeng\_db.teachers t

JOIN top\_teacher tt ON t.id\_teacher = tt.id\_teacher

-- where id\_teacher in (select id\_teacher from top\_teacher)

**SQL6. Подзапросы и WITH\_ДЗ 6. Задача 2**

WITH TMC as

(SELECT id\_teacher,

date\_trunc('month', class\_start\_datetime) AS MONTH,

count(id\_class) AS classes

FROM skyeng\_db.classes

WHERE class\_status in ('success',

'failed\_by\_student')

AND date\_trunc('year', class\_start\_datetime) = '2016-01-01'

GROUP BY date\_trunc('month', class\_start\_datetime),

id\_teacher

)

SELECT month, sum(classes)/count(id\_teacher) as class\_per\_teacher

FROM TMC

GROUP BY month

**SQL7. Оконные функции\_ДЗ 7. Задача 1**

WITH payment\_per\_day AS

(SELECT sum(payment\_amount) AS payment\_sum,

count(\*) AS payment\_count,

date\_trunc('day', transaction\_datetime) AS transaction\_date

FROM skyeng\_db.payments

WHERE id\_transaction IS NOT NULL

AND status\_name = 'success'

AND operation\_name in ('Покупка уроков',

'Начисление корпоративному клиенту')

GROUP BY date\_trunc('day', transaction\_datetime))

SELECT

transaction\_date,

sum(payment\_count) OVER (ORDER BY transaction\_date) cumulative\_count,

avg(payment\_sum) OVER (ORDER BY transaction\_date ROWS BETWEEN 3 PRECEDING AND 3 FOLLOWING) ma7days,

avg(payment\_sum) OVER (ORDER BY transaction\_date ROWS BETWEEN 15 PRECEDING AND 15 FOLLOWING) ma31days

FROM payment\_per\_day

ORDER BY transaction\_date

**SQL7. Оконные функции\_ДЗ 7. Задача 2**

SELECT

user\_id,

payment\_amount,

transaction\_datetime::date,

id\_transaction,

row\_number()

OVER (PARTITION BY user\_id ORDER BY transaction\_datetime) AS payment\_num,

row\_number()

OVER (PARTITION BY user\_id, date\_trunc('month', transaction\_datetime) ORDER BY transaction\_datetime) AS payment\_num\_month,

sum(payment\_amount)

OVER (PARTITION BY user\_id, date\_trunc('month', transaction\_datetime)) month\_payment,

lag(transaction\_datetime)

OVER (PARTITION BY user\_id ORDER BY transaction\_datetime) AS prev\_payment,

lead(transaction\_datetime)

OVER (PARTITION BY user\_id ORDER BY transaction\_datetime) AS next\_payment,

transaction\_datetime - lag(transaction\_datetime)

OVER (PARTITION BY user\_id ORDER BY transaction\_datetime) AS payment\_interval

FROM skyeng\_db.payments

WHERE id\_transaction IS NOT NULL

AND status\_name = 'success'

AND operation\_name in ('Покупка уроков',

'Начисление корпоративному клиенту')

ORDER BY transaction\_datetime

**SQL8. Создание, изменение, удаление данных в таблице, создание индексов + Временные таблицы\_ДЗ 8. Задача 1**

create temp table capitalization\_s1 as

select 'Россия' as country, 'Сбербанк' as company, 79504 as capitalization union all

select 'Россия', 'Газпром', 68012 union all

select 'Россия', 'Яндекс', 22122 union all

select 'Россия', 'Татнефть', 15176 union all

select 'Россия', 'X5 Retail Group', 9809 union all

select 'США', 'Apple', 1113000 union all

select 'Китай', 'Alibaba', 522000 union all

select 'США', 'Amazon', 1000000 union all

select 'Саудовская Аравия', 'Saudi Aramco', 1602000 union all

select 'США', 'Facebook', 475000;

-- select \* from capitalization\_s1

create temp table capitalization\_s2 as

select country, max(capitalization) as max\_capitalization

from capitalization\_s1

group by country

-- select \* from capitalization\_s2

create temp table capitalization\_s3 as

select s1.\*

from capitalization\_s1 s1

join capitalization\_s2 s2 on s2.max\_capitalization = s1.capitalization

-- select \* from capitalization\_s3

-- drop table capitalization\_s1

-- drop table capitalization\_s2

-- drop table capitalization\_s2

**SQL8. Создание, изменение, удаление данных в таблице, создание индексов + Временные таблицы\_ДЗ 8. Задача 2**

CREATE LOCAL TEMP TABLE top3host AS

SELECT l.host\_id,

count(r.\*)

FROM airbnb\_dallas.listings l

JOIN airbnb\_dallas.reviews r ON l.id = r.listing\_id

WHERE date\_trunc('month', r.date) = '01.12.2020'

GROUP BY l.host\_id

ORDER BY 2 DESC

LIMIT 3;

CREATE LOCAL TEMP TABLE top3price AS

SELECT l.host\_id,

max(l.price) AS max\_price

FROM airbnb\_dallas.listings l

JOIN top3host h USING(host\_id)

GROUP BY l.host\_id;

SELECT l.host\_id,

l.name,

l.price

FROM airbnb\_dallas.listings l

JOIN top3price p ON l.host\_id = p.host\_id

AND l.price = p.max\_price;

**SQL8. Создание, изменение, удаление данных в таблице, создание индексов + Временные таблицы\_ДЗ 8. Задача 3**

CREATE LOCAL TEMP TABLE payments\_with\_sex AS

SELECT s.user\_id,

s.student\_sex,

count(p.\*) AS payments

FROM skyeng\_db.payments p

JOIN skyeng\_db.students s ON p.user\_id = s.user\_id

WHERE p.operation\_name in ('Покупка уроков',

'Начисление корпоративному клиенту')

AND p.status\_name = 'success'

AND s.student\_sex IS NOT NULL

GROUP BY s.user\_id,

s.student\_sex;

SELECT

(SELECT count(\*) FROM payments\_with\_sex WHERE student\_sex = 'f' AND payments = 2)/

(SELECT count(\*) FROM payments\_with\_sex WHERE student\_sex = 'f' AND payments > 2)::float AS female,

(SELECT count(\*) FROM payments\_with\_sex WHERE student\_sex = 'm' AND payments = 2)/

(SELECT count(\*) FROM payments\_with\_sex WHERE student\_sex = 'm' AND payments > 2)::float AS male;

**SQL10. Основы оптимизации\_ДЗ 10. Задача 1**

explain

with teachers\_cost as

(

select id\_teacher,

case when language\_group = 'rus' then 900 else 1500 end as class\_cost

from skyeng\_db.teachers

)

select

date\_trunc('month', class\_start\_datetime) as class\_month,

sum(class\_cost) as total\_classes\_cost, count(id\_class) as classes\_count,

sum(class\_cost)::float / count(id\_class) as avg\_cost

from skyeng\_db.classes c

left join teachers\_cost on teachers\_cost.id\_teacher = c.id\_teacher

where class\_status in ('success', 'failed\_by\_teacher')

-- and class\_start\_datetime >= '2016-01-01'::timestamp -- все данные и так удовлетворяют этому условию

and class\_start\_datetime < '2017-01-01'::timestamp

and class\_type != 'trial'

group by 1

order by 1 --если убрать, то снизит cost на ~1000, но требуется для удобства