Занятие 22

Pandas, SQL

Что напечатают эти операторы?

```
lst = ['a', 'b', 'c', 'd', 'e']
print( lst[10:])

lst = [ [ ] ] * 5
print(lst)

lst[0].append(10)
print(lst)

lst[1].append(20)
print(lst)

lst.append(30)
print(lst)
```

Задача 21-1

Составьте в произвольной форме список полей в таблице Pupil, предполагая, что она будет использоваться в системе учета и планирования курса «Разработчик на Питоне»

Задача 21-2

Напишите функцию, которая находит оптимальный маршрут из верхнего левого угла в правый нижний угол матрицы. Двигаться можно только вправо или вниз. В каждой клетке матрице содержится число. Оптимальным считается маршрут с минимальной суммой чисел клеток, через которые проходит маршрут.

Например, если матрица 3 x 3:

10	20	30
5	1	80
90	2	70

, то оптимальным будем маршрут, который проходит через клетки 10 + 5 + 1 + 2 + 70 = 88.

Подсказка: полный перебор вариантов заведомо неэффективен.

Возможное решение – идти от начала и запоминать оптимальные маршруты для каждой клетки из начала.

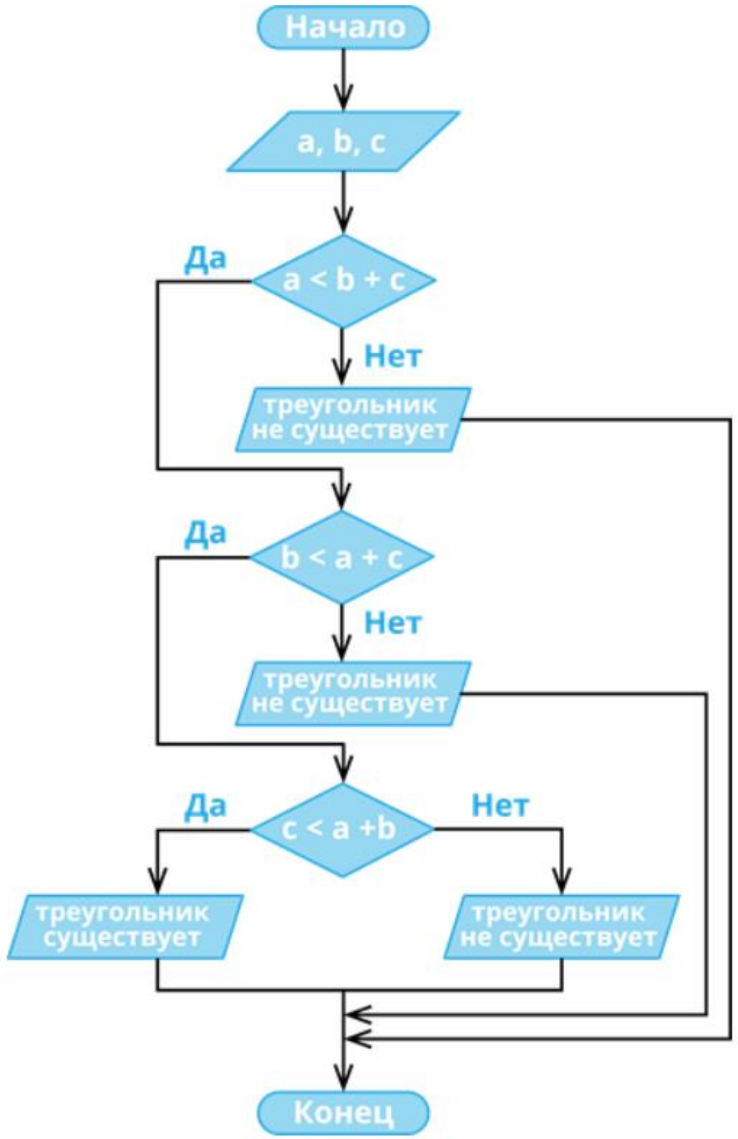
Задача 21-3

Создайте таблицу book с полями book_id (int), book_title (text), book_author (text), publisher_id(int).

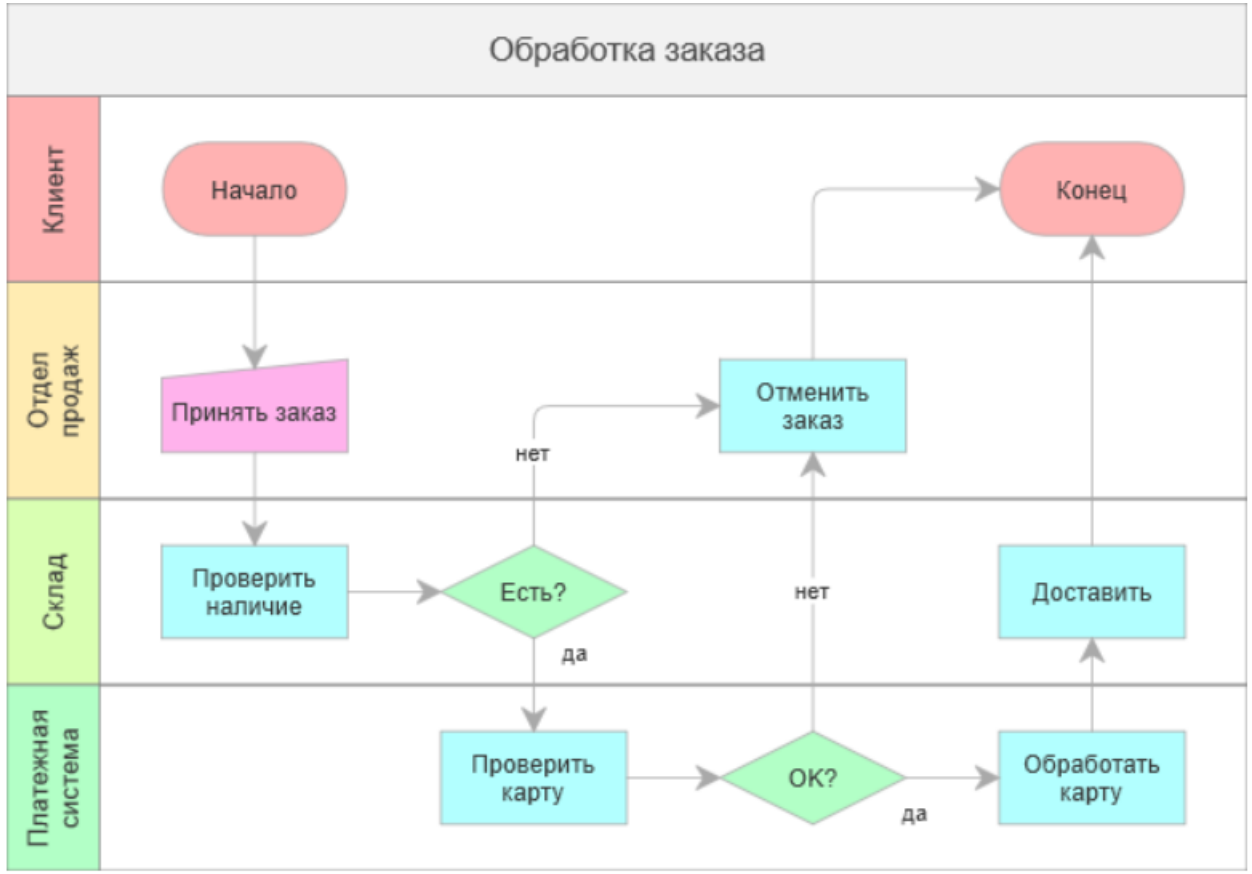
Введите несколько книг нескольких авторов.

Сделайте несколько SELECT для выборки книг по авторам, по названиям.

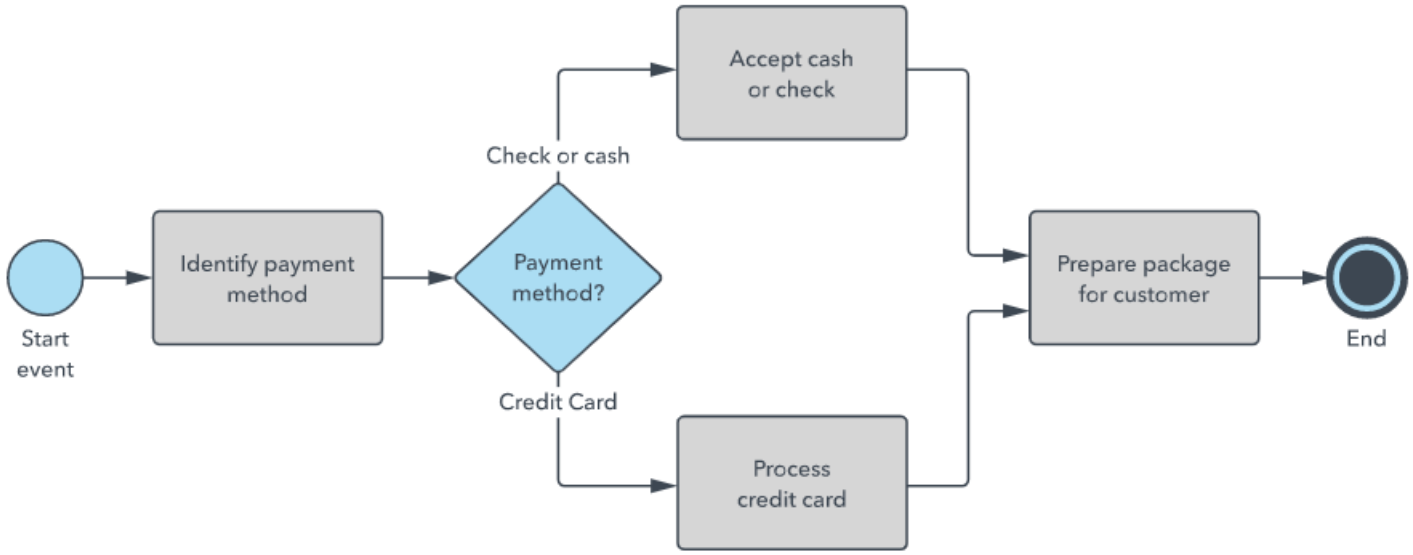
Блок-схемы алгоритмов



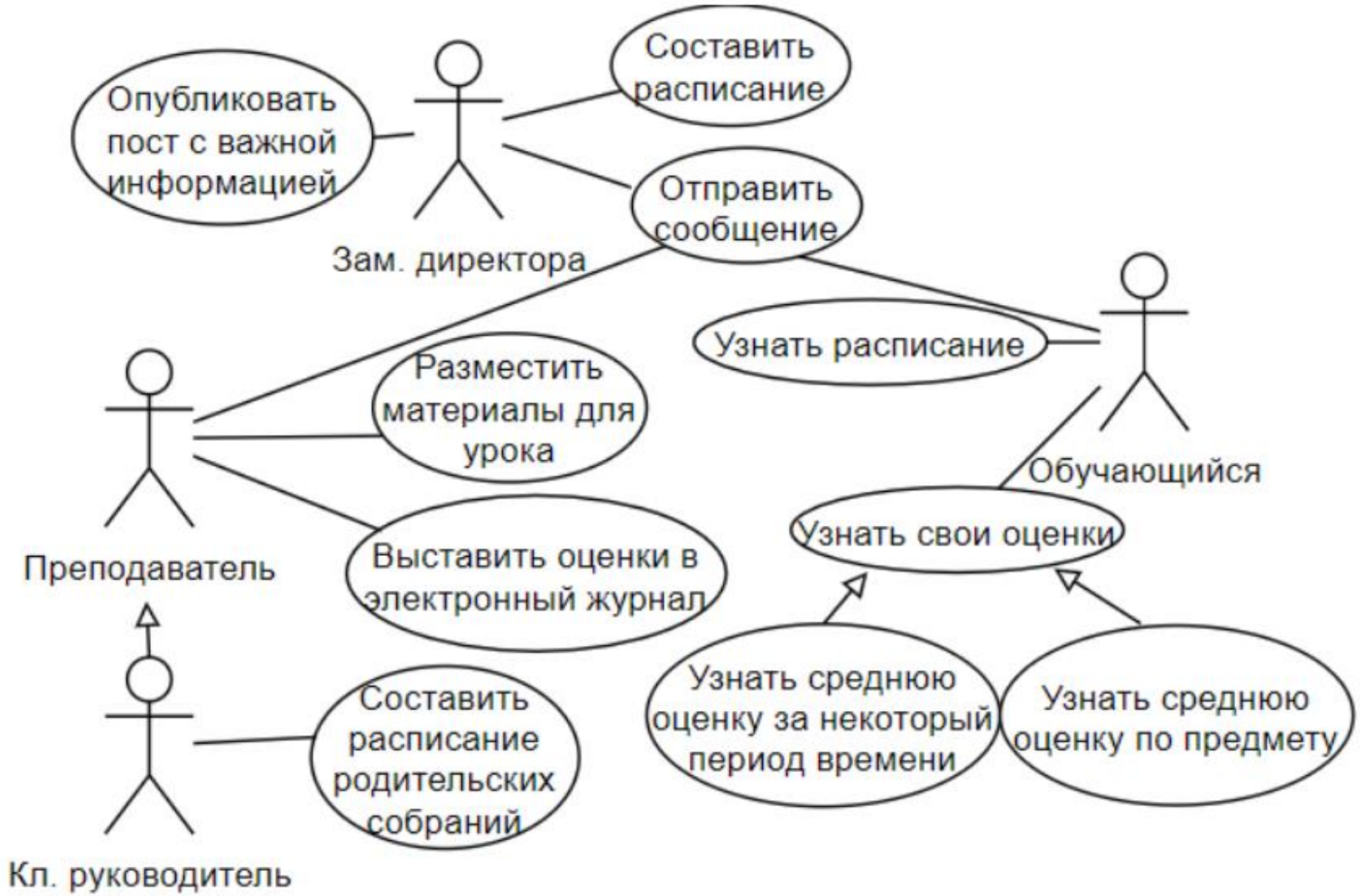
swim-lane диаграммы



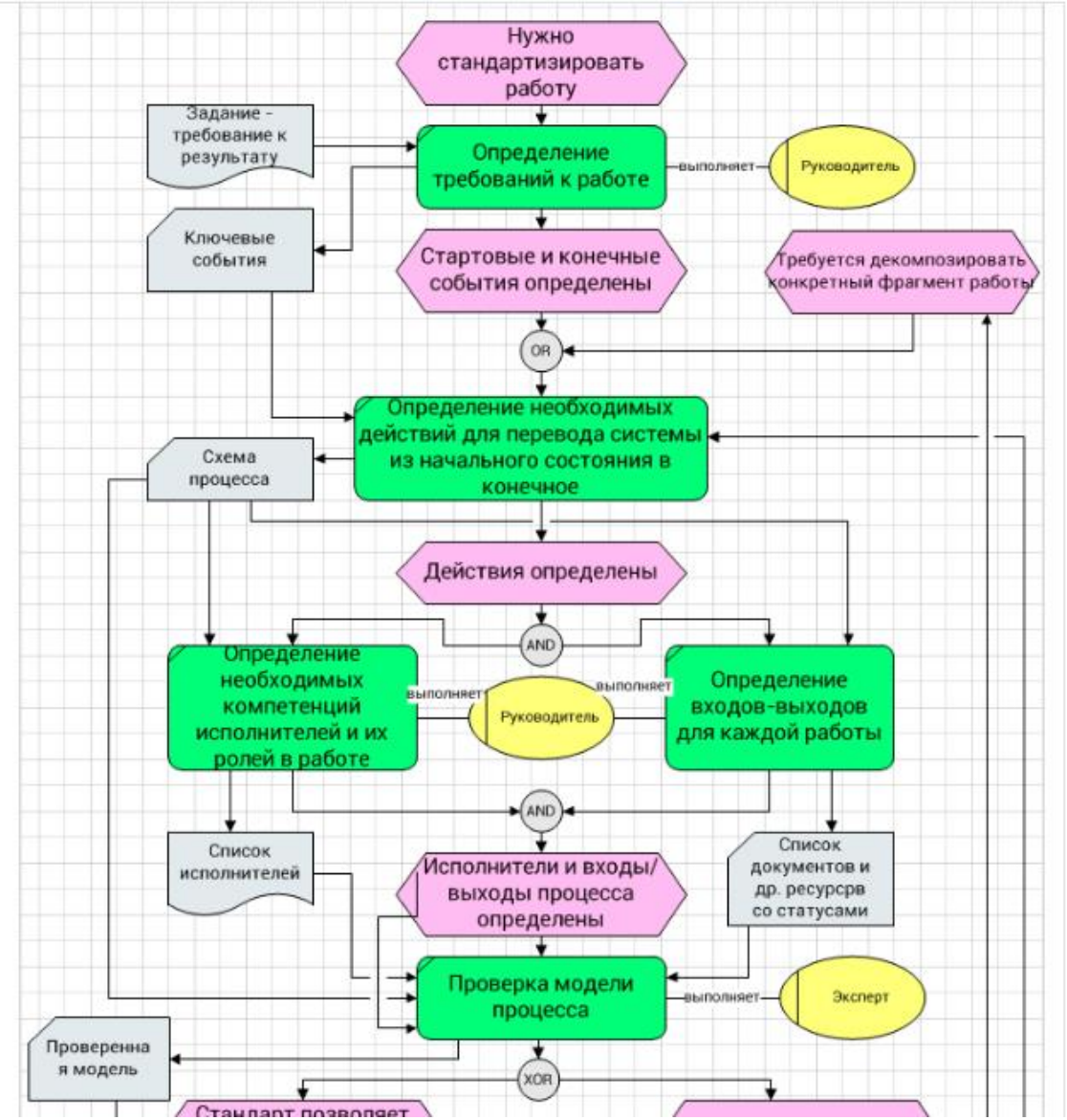
Business Process Model and Notation

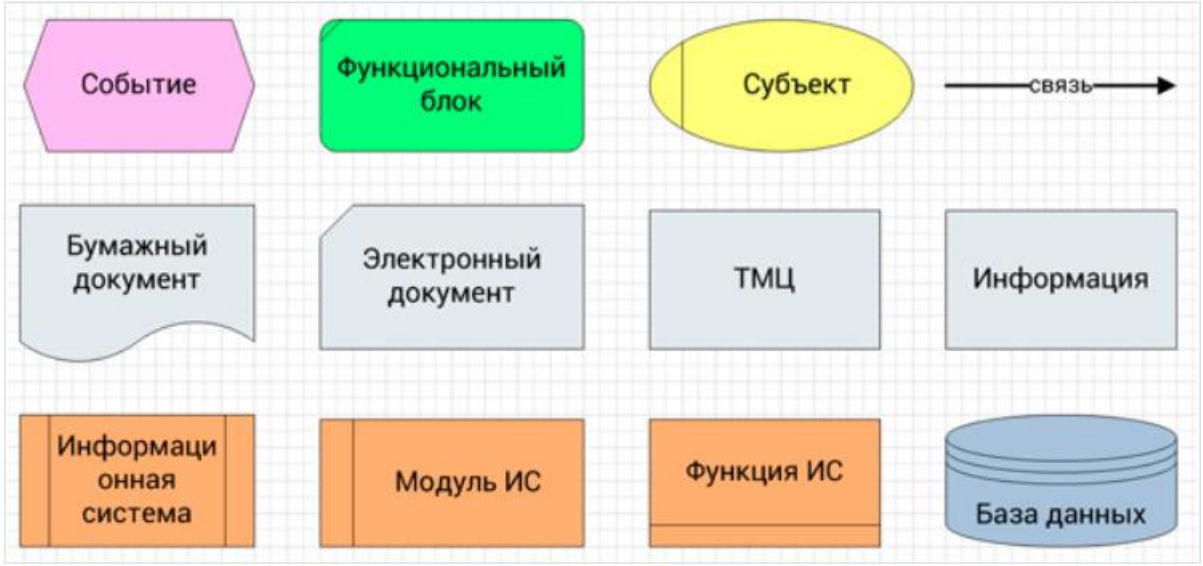


Use-case диаграммы



EPC (Event-driven Process Chain) диаграммы





Pandas

Pandas — это библиотека Python для обработки и анализа структурированных данных, её название происходит от «panel data» («панельные данные»).

csv, excel

Импорт:

```
df_from_csv = pd.read_csv(‘?????/anime.csv’)
df_from_excel = pd.read_excel(‘?????/test.xlsx’)
```

Экспорт:

```
rating[:10].to_csv('saved_ratings.csv', index=False)
df.to_excel
```

Работа с данными

```
df[‘a1’] # выбор одной колонки
df[[‘a1’, ‘b2’]]  # выбор нескольких колонок
df.loc[0] # выбор одной строки
df.loc[0:2] # выбор нескольких строк
df[ df[‘Цена’] > 10] # выбор по условию
df[ (df[‘a1’] = ‘x’) | (df[‘a1’[ = ‘y’)] # выбор по условию
df.loc[‘a’]
df.iloc[0]
df[0:3]
```

```
New_df = df.T # Транспонирование
df.loc[0, 'Шт'] = 123
df.index
df.columns
df1 = df.set_index('Год')
df1.index
df1.loc[2002:2003]
df1.iloc[1:3]
df1.loc[2007] = ['F', 30, 100, 3000]
```

Выборки, информация

```
anime.head(3)          # первые три строки
rating.tail(2)         # последние две
len(df)          # количество строк в df
len(ratings['user_id'].unique()) # количество уникальных значений в столбце
 df.describe()         # статистика о df
 anime.col1.value_counts() # количество значений в конкретном столбце
 df.columns            # список наименований колонок
 df.index              # список индексов
 df.loc[2000]          # конкретная строка
 df.sum()
 print(df1[['Год', 'Цена’]].sum())
 df.mean()
print(df1[['Год', 'Цена']].mean())
```

Изменение данных

```
anime['train set'] = True  # добавить новый столбец с конкретным значением
df_new = df[[‘a1’, ‘b2’]]   # новый дейта-фрейм из подмножества столбцов


df1 = anime[0:2]
df2 = anime[2:4]
pd.concat([df1, df2], ignore_index=True)


df[‘new’] = df[‘a1’] + df[‘b2’] # Новая колонка
df + df # Что получится?
```

Фильтрация

```
anime_modified.iloc[0:3] # первые три строчки, используется номер строки
anime[anime[‘type’] == 'TV’] # выбрать строчки по одном значению
anime[anime['type'].isin(['TV', 'Movie’])] # выбрать строчки по нескольким значениям
anime[anime['rating'] > 8]          # фильтрация по значению


anime.sort_values('rating', ascending=False) # сортировка


anime.groupby('type').count() # подсчитать сколько строчек с каждым значением в столбце
df.loc[(df.age > 50) & (df.gender == 'Female’)]
df.loc[(df.country == 'Poland') | (df.country == 'Czech Republic')]
```

Задания

1. Найдите максимальную сделку

```
ma = df['Итого'].max()
df[df['Итого'] == df['Итого'].max()]
```

2. Напечатать каждую вторую строку DataFrame

```
df[1:len(df):2]
```

3. Напечатать все строки, у которых цена меньше средней

```
me = df['Цена'].mean()
print(df[df['Цена'] < me])
```

4. Дан df. Найдите минимальное значение для каждого столбца. Потом минимальное значение среди всех строковых значений и минимальное значение для всех числовых значений

```
mi = []
for i in df.columns:
    mi.append(df[i].min())
mi_number = min(i for i in mi if isinstance(i,(int, float)))
mi_str = min(i for i in mi if type(i) == str)
print(mi, mi_number, mi_str)
```

У нас есть таблица продаж. Запросы к ней.

1. Три самые большие сделки

```
df.sort_values('Итого', ascending = False).head(3)
```

2. Три самые маленькие сделки

```
df.sort_values('Итого', ascending = True).head(3)
```

3. Суммы сделок для каждого товара

```
df.groupby('Товар').Итого.sum()
```

4. Наибольшее число из них

```
df.groupby('Товар').Итого.sum().max()
```

5. Суммы сделок для каждого года

```
df.groupby('Год').Итого.sum()
```

6. Наибольшее число из них

```
df.groupby('Год').Итого.sum().max()
```

7. Продажи товаров по штукам

```
df.groupby('Товар').Шт.sum()
```

8. Сумма продаж в конкретном году

```
df[df['Год'] == 2003].Итого.sum()
```

join

```
a = {'Год':[2001, 2002, 2003, 2004], 'Шт':[1,2,3, 4], 'A':[100, 200, 300, 400]}
df = pd.DataFrame(a, index = [2001, 2002, 2003, 2004])
df1 = pd.DataFrame({'Company':['aa', 'bb', 'cc', 'dd']}, index = [2001, 2002, 2003, 2004])
df2 = df.join(df1)
print(df2)
```

	Год	Шт	A	Company
2001	2001	1	100	aa
2002	2002	2	200	bb
2003	2003	3	300	cc
2004	2004	4	400	dd

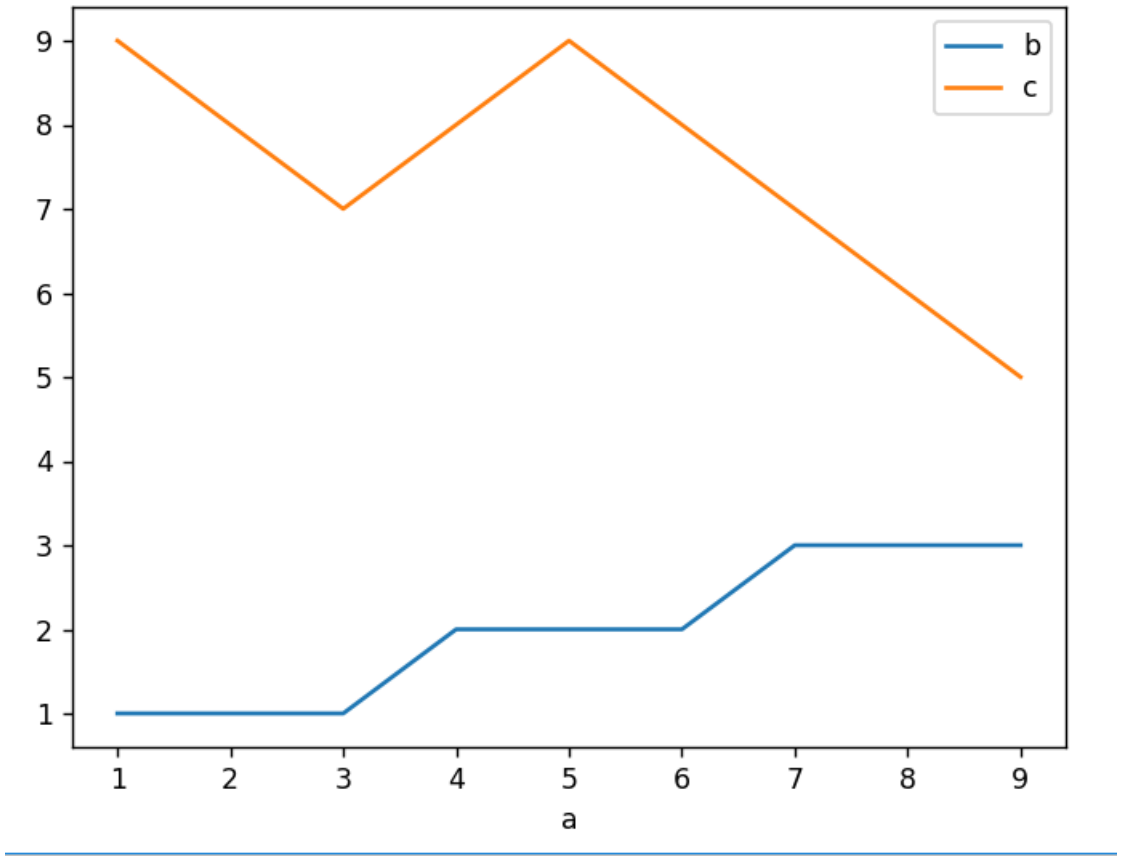
concat

```
a = {'Год':[2001, 2002, 2003, 2004], 'Шт':[1,2,3, 4], 'A':[100, 200, 300, 400]}
df = pd.DataFrame(a, index = [2001, 2002, 2003, 2004])
df1 = df.loc[[2001, 2002]]
df2 = df.loc[[2003, 2004]]
df3 = pd.concat([df1, df2])
 print(df3)
```

	Год	Шт	A
2001	2001	1	100
2002	2002	2	200
2003	2003	3	300
2004	2004	4	400

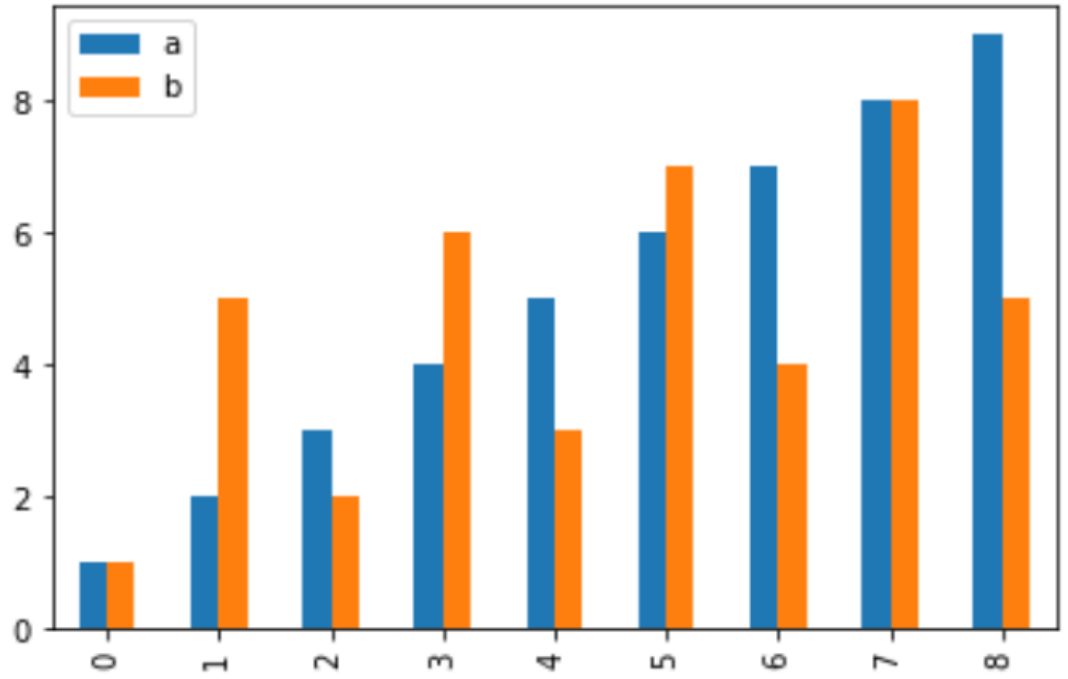
Визуализация

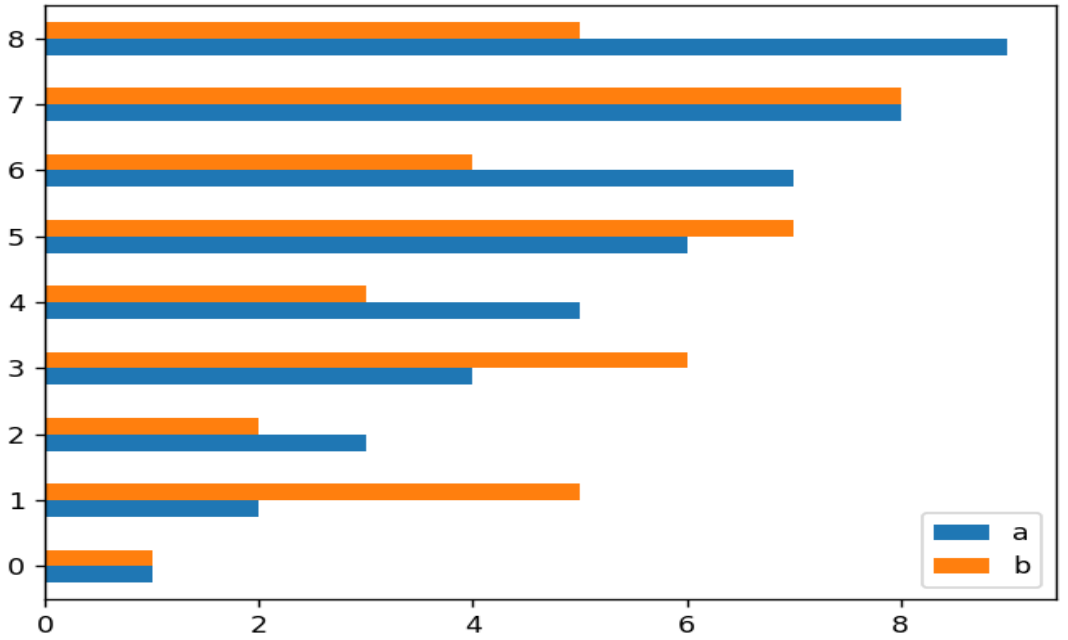
```
import matplotlib.pyplot as plt
import pandas as pd
df = pd.DataFrame({'a':[1,2,3,4,5,6,7,8,9],
                   'b':[1,1,1,2,2,2,3,3,3],
                   'c':[9,8,7,8,9,8,7,6,5] })
df.index = df['a']
df[['b', 'c']].plot()
plt.show()
```



Гистограммы

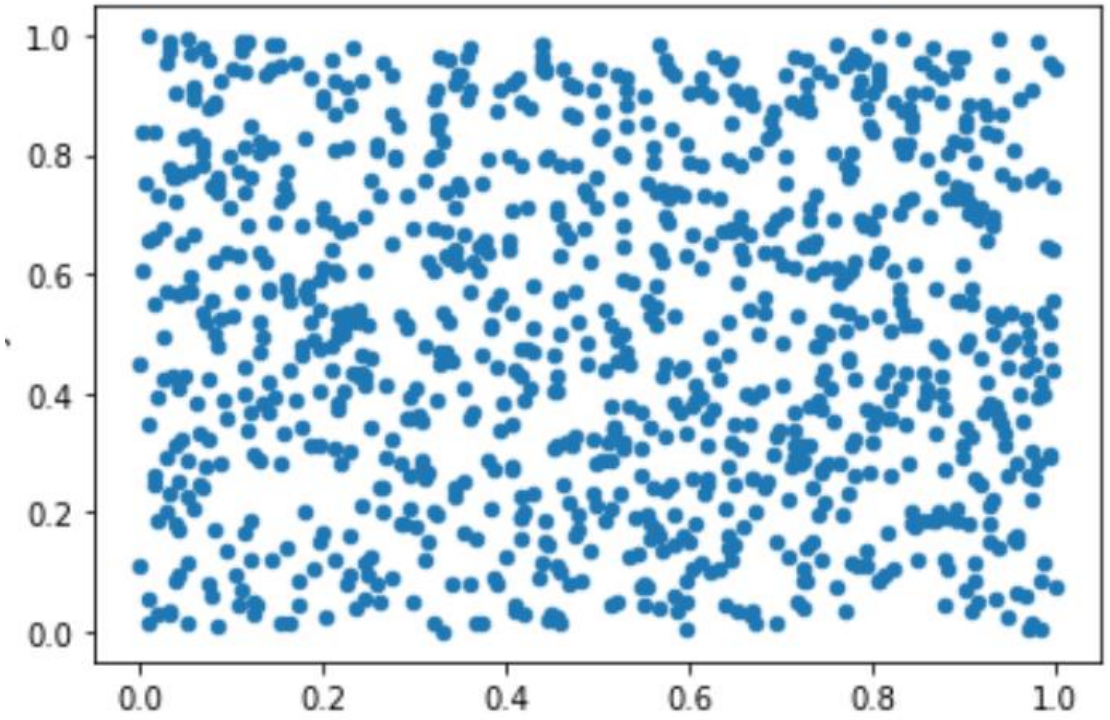
```
import matplotlib.pyplot as plt
import pandas as pd


df = pd.DataFrame({'a':[1,2,3,4,5,6,7,8,9],
'b':[1,5,2,6,3,7,4,8,5]})
df.plot.barh()
#df.plot.bar()
plt.show()
```





Распределение точек

```
import matplotlib.pyplot as plt
import pandas as pd
import random
x, y = [], []
for _ in range(1000):
    x.append(random.random())
    y.append(random.random())
df = pd.DataFrame({'x':x, 'y':y})
df.plot('x','y', kind = 'scatter')
plt.show()
```



Задание

Создайте DataFrame состоящий из значений аргументов от -10 до 10 (столбец ‘x’) и значений их кубов (столбец ‘y’).

Нарисуйте график функции, используя matplotlib.

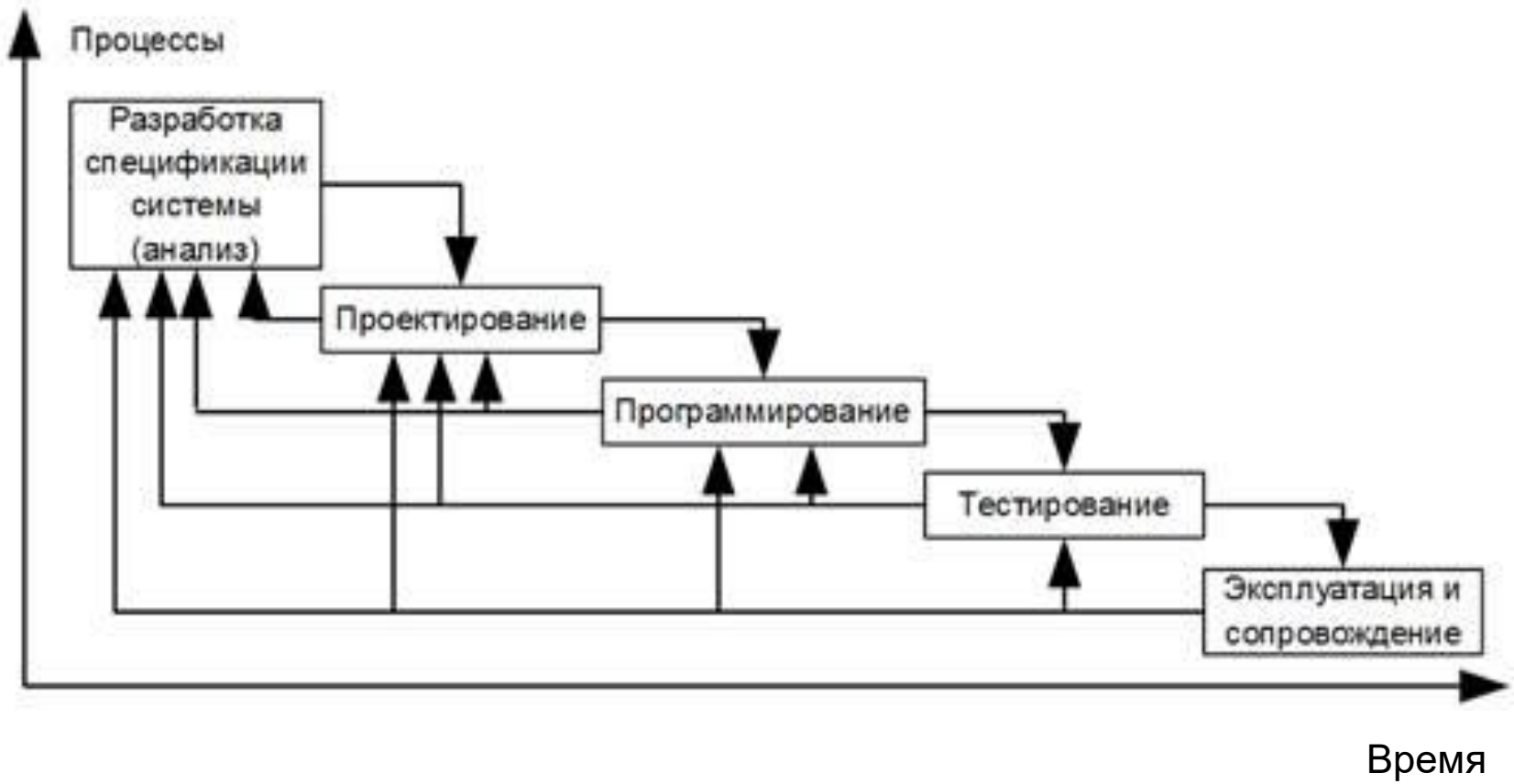
DB Design


Data Management
Data Capture
Data Migration
Database Development
Database Management
Data Maintenance
Database Query
Statistical Analysis
Relational Database
Data Analysis

Автоматизированная информационная система (АИС)

- АИС - это совокупность программных и аппаратных средств, предназначенных для хранения и/или управления данными и информацией, а также для производства вычислений.
- АИС представляют совокупность функциональных подсистем сбора, ввода, обработки, хранения, поиска и распространения информации. Процессы сбора и ввода данных необязательны, поскольку вся необходимая и достаточная для функционирования АИС информация, может уже находиться в составе ее базы данных. Каждая АИС имеет дело с той или иной частью реального мира, которую принято называть предметной областью

Жизненный цикл разработки ПО (АИС,ИС ...)



Предметная область

- Это сфера экономической деятельности, например, транспортная логистика, банковский сектор, биллинговые системы, медицина, электронная коммерция, игровое приложение и все остальные.

База данных (БД)

- Именованная совокупность данных, отражающая состояние объектов и их отношений в рассматриваемой предметной области, которая допускает использование данных оптимальным образом для одного или нескольких приложений.
- В БД информация должна быть организована таким образом, чтобы обеспечить минимальную долю ее избыточности. Частичная избыточность информации необходима, но она должна быть минимизирована, т.к. чрезмерная избыточность данных влечет за собой ряд негативных последствий, главные из которых:
- увеличение объема информации
- появление ошибок при вводе дублирующей информации, нарушающих целостность базы данных и создающих противоречивые данные.

Для организации БД используют СУБД

- **Реляционные** - совокупность таблиц и связей между ними. Примеры: PostgreSQL, SQLite
- **Документно-ориентированные** - хранит иерархические структуры данных (документы), как правило реализована с помощью подхода NoSQL. Примеры: MongoDB, CouchDB
- **Объектно-ориентированные** - хранят информацию в виде объектов, как в объектно-ориентированных языках программирования. Примеры: db4o
- **Графовые** - предназначены для хранения взаимосвязей и навигации в них, данные хранятся в виде структуры данных граф, где узлы хранят сущности, а ребра связи. Примеры: OrientDB, Amazon Neptune, Neo4j,

Система управления базами данных (СУБД)

- Совокупность языковых и программных средств, предназначенных для создания, ведения и совместного использования БД многими пользователями. возможности современных СУБД:
- язык определения данных(DDL), с помощью которого можно создать базу данных, описать ее структуру, типы данных и средства для задания ограничений целостности данных;
- язык управления данными(DML), которые позволяет вставлять, обновлять, удалять и извлекать информацию из БД;

Возможности СУБД

- кроссплатформенность - независимость от используемой архитектуры или ОС;
- подсистема разграничения доступа к данным;
- подсистема поддержки целостности БД обеспечивающая непротиворечивое состояние хранимых данных;
- подсистема управления параллельной работой приложений контролирующая процессы их совместного доступа к БД;
- подсистема восстановления, позволяющая вернуть БД к предыдущему состоянию, нарушенному из-за аппаратного или программного сбоя;

Реляционные базы данных как один из типов БД

- **Реляционная база данных** – это совокупность отношений, содержащих всю информацию, которая должна храниться в БД. Однако пользователи могут воспринимать такую базу данных как совокупность таблиц.

- **Моделирование данных** - это средство формального сбора данных, относящихся к бизнес-процессу данной организации. Моделирование является приемом анализа, на базе которого строятся реляционные БД.

DB Design

- Процесс моделирования данных включает три уровня моделей: от концептуальной к логической, а затем к физической.
- **Концептуальная модель** - модель предметной области, состоящая из перечня взаимосвязанных объектов, используемых для описания этой области, вместе со свойствами и характеристиками.

Концептуальная модель, как правило, представляет сущность бизнеса и ничего больше.

- **Логическая модель** - представляется диаграммой «сущность-связь» или ER (Entity-Relationship) диаграммой.
- **Физическая модель** - это схема базы данных для конкретной СУБД, где сущности предметной области превращаются в таблицы, атрибуты в ее колонки (часто называют полями) с использованием конкретного типа данных, связи в ограничения целостности.

Концептуальная модель данных

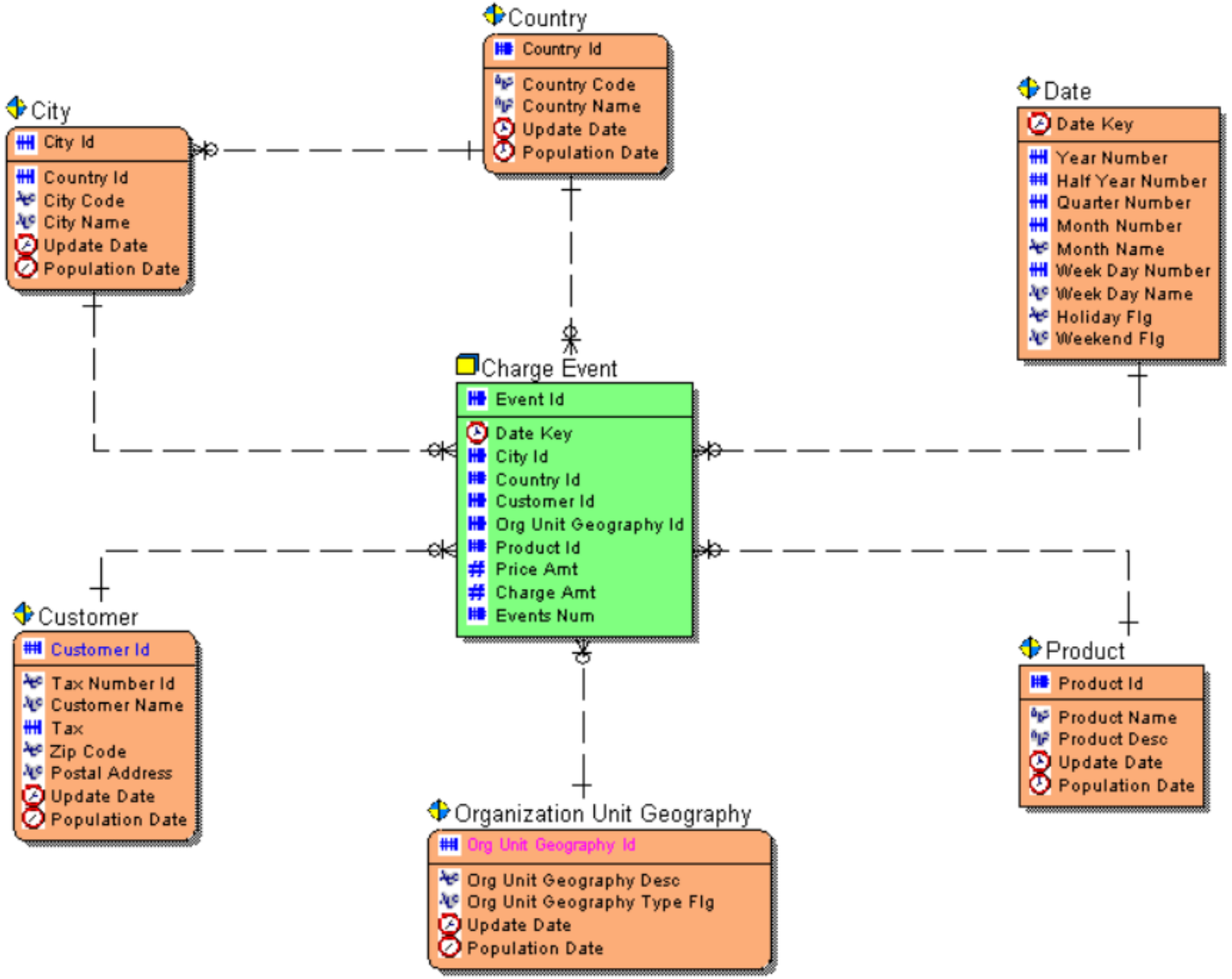
- Концептуальная модель данных (КМД) – это общая информационная модель предметной области, охватывающая вопросы классификации, структуризации и семантической целостности (достоверности и согласованности данных).
- Концептуальная модель данных разрабатывается независимо от ограничений, вытекающих из моделей данных, поддерживаемая той или иной СУБД.
- Для описания концептуальной модели может быть использован естественный язык, но такое описание будет громоздким и неоднозначным, поэтому для описания КМ чаще всего используется формализованный язык.

Пример модели - магазин

Логическое моделирование

- На этапе логического моделирования выявляются сущности, их атрибуты и связи между ними.
- **Сущность** - это объект реального мира, который может быть как материальным, так и нет. Пример материальных объектов: покупатель, продукт, автомобиль. Пример нематериальных объектов: заказ, формула, расписание.
- С логической точки зрения сущность представляет собой совокупность однотипных объектов называемых экземплярами этой сущности. Экземпляры сущности должны быть уникальными, то есть полный набор значений их атрибутов не должен дублироваться.

Пример модели

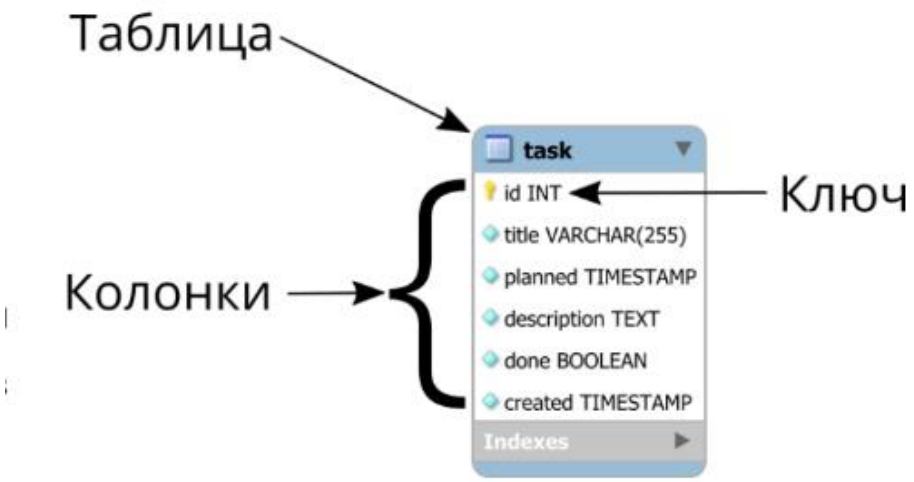


Атрибуты

- **Атрибуты сущности** - это фактически свойства объекта. Например, у покупателя есть фамилия, имя и отчество - это его свойства или атрибуты.
- **Атрибуты** могут быть ключевыми и не ключевыми. Например, у покупателя может быть **уникальный идентификатор**, значение которого можно использовать в атрибуте “покупатель” сущности заказ.
- На этапе логического проектирования для каждого атрибута обычно определяется примерный тип данных (строковый, числовой, логический, двоичные данные и др.).

Физическая модель

- Физическая модель - это схема базы данных для конкретной СУБД, где сущности предметной области превращаются в таблицы, атрибуты в ее колонки (часто называют столбцами) с использованием конкретного типа данных, связи в ограничения целостности. Строка таблицы - экземпляр одной сущности предметной области.



Физическая модель

Специфика конкретной СУБД при физическом проектировании включает выбор решений, связанных с физической средой хранения данных.

Выбор методов управления дисковой памятью, разделение БД по файлам и устройствам, методов доступа к данным, создание индексов, и т.д.

Модифицировать таблицу

```
ALTER TABLE book ADD PRIMARY KEY (book_id)
ALTER TABLE book ADD COLUMN abc int
UPDATE book SET abc = 1 WHERE book_id = 1
UPDATE book SET abc = price * amount
ALTER TABLE book8 DROP COLUMN abc
```

book_id	title	author	price	amount
INT PRIMARY KEY	VARCHAR(50)	VARCHAR(30)	INT	INT
1	Мастер и Маргарита	Булгаков М.А.	670	3
2	Белая гвардия	Булгаков М.А.	540	5
3	Идиот	Достоевский Ф.М.	460	10
4	Братья Карамазовы	Достоевский Ф.М.	799	2
5	Стихотворения и поэмы	Есенин С.А.	650	15

SELECT

SELECT title, amount FROM book;

Использовать названия

SELECT title AS Название, amount FROM book;

Вычисляемые поля

SELECT title, author, price, amount, price * amount AS total FROM book;

Функции

Функция	Описание	Пример
CEILING(x)	возвращает наименьшее целое число, большее или равное **x** (округляет до целого числа в большую сторону)	CEILING(4.2)=5 CEILING(-5.8)=-5
ROUND(x, k)	округляет значение **x** до **k** знаков после запятой, если **k** не указано – **x** округляется до целого	ROUND(4.361)=4 ROUND(5.86592,1)=5.9
FLOOR(x)	возвращает наибольшее целое число, меньшее или равное **x** (округляет до целого числа в меньшую сторону)	FLOOR(4.2)=4 FLOOR(-5.8)=-6
POWER(x, y)	возведение **x** в степень **y**	POWER(3,4)=81.0
SQRT(x)	квадратный корень из **x**	SQRT(4)=2.0 SQRT(2)=1.41...
DEGREES(x)	конвертирует значение **x** из радиан в градусы	DEGREES(3) = 171.8...
RADIANS(x)	конвертирует значение **x** из градусов в радианы	RADIANS(180)=3.14...
ABS(x)	модуль числа **x**	ABS(-1) = 1 ABS(1) = 1
PI()	pi = 3.1415926...	

Задание

Сосчитайте налоги и цену с налогом

```
SELECT title, price, (price*20/100) AS tax, price * 1.2 AS price_tax FROM book;
```

Округлите

```
SELECT title, price, ROUND((price*20/100, 2) AS tax, ROUND(price*1.2,2) AS price_tax FROM book;
```

В конце года цену всех книг на складе пересчитывают – снижают ее на 30%.

Написать SQL запрос, который из таблицы book выбирает названия, авторов, количества и вычисляет новые цены книг.

Столбец с новой ценой назвать new_price, цену округлить до 2-х знаков после запятой

Where

Выборка, если цена меньше 600

- SELECT title, price FROM book WHERE price < 600;

Полная стоимость, если цена больше 4000

- SELECT title, author, price * amount AS total FROM book WHERE price * amount > 4000;

Книги Булгакова, если цена больше 600

- SELECT title, author, price FROM book WHERE price > 600 AND author = 'Булгаков М.А.';

Или Булгаков, или Есенин и цена больше 600

- SELECT title, author, price FROM book WHERE (author = 'Булгаков М.А.' OR author = 'Есенин С.А.') AND price > 600;

Between, In, Order By, Asc, Desc

```
SELECT title, amount FROM book WHERE amount BETWEEN 5 AND 14;
```

То же самое:

```
SELECT title, amount FROM book WHERE amount >= 5 AND amount <=14;
```

```
SELECT title, price FROM book WHERE author IN ('Булгаков М.А.', 'Достоевский Ф.М.’);
```

```
SELECT title, author, price FROM book ORDER BY title;
```

```
SELECT title, author, price FROM book ORDER BY 1;
```

```
SELECT author, title, amount AS Количество FROM book WHERE price < 750 ORDER BY author, amount DESC;
```

```
select author, title from book where amount between 2 and 14 order by author desc, title asc
```

LIKE

Символ-шаблон	Описание	Пример
%	Любая строка, содержащая ноль или более символов	SELECT * FROM book WHERE author LIKE '%M.%' выполняет поиск и выдает все книги, инициалы авторов которых содержат «*M.*»
_ (подчеркивание)	Любой одиночный символ	SELECT * FROM book WHERE title LIKE 'Поэм_' выполняет поиск и выдает все книги, названия которых либо «*Поэма*», либо «*Поэмы*» и пр.

```
SELECT title FROM book WHERE title LIKE 'Б%’;
SELECT title FROM book  WHERE title LIKE "_____“
SELECT title FROM book WHERE title LIKE "______%";
```

Вывести названия книг, которые состоят ровно из одного слова, если считать, что слова в названии отделяются друг от друга пробелами

```
SELECT title FROM book  WHERE title NOT LIKE '% %';
```

Distinct, Sum, Count, Group By

```
SELECT DISTINCT author FROM book;
SELECT author, count(amount) FROM book GROUP BY author;
SELECT author, SUM(amount) FROM book GROUP BY author;
```

MIN, MAX, AVERAGE

```
SELECT author, MIN(price) AS min_price FROM book GROUP BY author;
```

```
select author, min(price) as Минимальная_цена, max(price) as Максимальная_цена, avg(price) as Средняя_цена
```

```
from book group by author
```

```
SELECT author, SUM(price * amount) AS Стоимость FROM book
GROUP BY author;
```

```
SELECT author, ROUND(AVG(price),2) AS Средняя_цена FROM book
GROUP BY author;
```

```
SELECT SUM(amount) AS Количество FROM book;
```

```
SELECT SUM(amount) AS Количество, SUM(price * amount) AS Стоимость FROM book;
```

Задание

Найти минимальную и максимальную цену книг всех авторов, общая стоимость книг которых больше 5000.

- SELECT author, MIN(price) AS Минимальная_цена, MAX(price) AS Максимальная_цена FROM book GROUP BY author HAVING SUM(price * amount) > 5000;

Найти минимальную и максимальную цену книг всех авторов, общая стоимость книг которых больше 5000. Результат вывести по убыванию минимальной цены.

- SELECT author,
- MIN(price) AS Минимальная_цена,
- MAX(price) AS Максимальная_цена
- FROM book
- GROUP BY author
- HAVING SUM(price * amount) > 5000
- ORDER BY Минимальная_цена DESC;

HAVING

Сначала определяется таблица, из которой выбираются данные (FROM), затем из этой таблицы отбираются записи в соответствии с условием WHERE, выбранные данные агрегируются (GROUP BY), из агрегированных записей выбираются те, которые удовлетворяют условию после HAVING.

Потом формируются данные результирующей выборки, как это указано после SELECT (вычисляются выражения, присваиваются имена и пр.). Результирующая выборка сортируется, как указано после ORDER BY.

```
SELECT author, MIN(price) AS Минимальная_цена, MAX(price) AS
Максимальная_цена
FROM book
WHERE author <> 'Есенин С.А.’
GROUP BY author
HAVING SUM(amount) > 10;
```

Вложенные запросы

```
SELECT title, author, price, amount
FROM book
WHERE price = ( SELECT MIN(price) FROM book );
```

```
select author, title, price
from book
where price <= (select avg(price) from book)
order by price DESC
```

Вложенные запросы

Вывести информацию о книгах, количество экземпляров которых отличается от среднего количества экземпляров книг на складе более чем на 3. То есть нужно вывести и те книги, количество экземпляров которых меньше среднего на 3, или больше среднего на 3.

```
SELECT title, author, amount
FROM book
WHERE ABS(amount - (SELECT AVG(amount) FROM book)) >3;
```

Вложенные запросы

Вывести информацию о книгах тех авторов, общее количество экземпляров книг которых не менее 12.

```
SELECT title, author, amount, price
FROM book
WHERE author IN (
    SELECT author
    FROM book
    GROUP BY author
    HAVING SUM(amount) >= 12
);
```

Вложенные запросы

Вывести информацию о книгах, количество экземпляров которых отличается от среднего количества экземпляров книг на складе более чем на 3, а также указать среднее значение количества экземпляров книг.

```
SELECT title, author, amount,
    (
     SELECT AVG(amount)
     FROM book
    ) AS Среднее_количество
FROM book
WHERE abs(amount - (SELECT AVG(amount) FROM book)) >3;
```

Задание

Создайте таблицу учеников, в которой хранятся средний балл выполненных заданий (от 1 до 5) и процент посещений занятий.

Придумайте и реализуйте запросы, которые показывают лучших и худших учеников по интегрированному показателю успеваемости и посещений (например, используя весовые коэффициенты).

Задача 22-1

Вывести содержимое таблицы book, при этом авторов отсортировать по возрастанию, а цены книг по убыванию

Задача 22-2

Дана структура типа бинарное дерево. Все вершины пронумерованы от 1 до n. Дерево задано в виде списка кортежей: [(1, 2), (1, 3), (2, 4), (2, 5), (3, 6), (6, 7), (7, 8)]

Каждый кортеж (a, b) показывает, что вершина a соединена с вершиной b.

По определению в дереве невозможны циклы.

Найти максимальную длину от вершины (1) до некоторой конечной вершины.

Задача 22-3

В Python существуют ключевые слова, которые нельзя использовать для названия переменных, функций и классов. Для получения списка всех ключевых слов можно воспользоваться атрибутом kwlist из модуля keyword.

Приведенный ниже код:

```
import keyword
print(keyword.kwlist)
```

выводит:

```
['False', 'None', 'True', 'and', 'as', 'assert', 'async', 'await', 'break', 'class', 'continue', 'def', 'del', 'elif', 'else', 'except', 'finally', 'for', 'from', 'global', 'if', 'import', 'in', 'is', 'lambda', 'nonlocal', 'not', 'or', 'pass', 'raise', 'return', 'try', 'while', 'with', 'yield']
```

Напишите программу, которая принимает строку текста и заменяет в ней все ключевые слова на <kw>