Задачи, решаемые информационными системами (ИС)

В широком понимании ИС — это любая система обработки информации, например, системы счисления, компьютерная сеть Интернет.

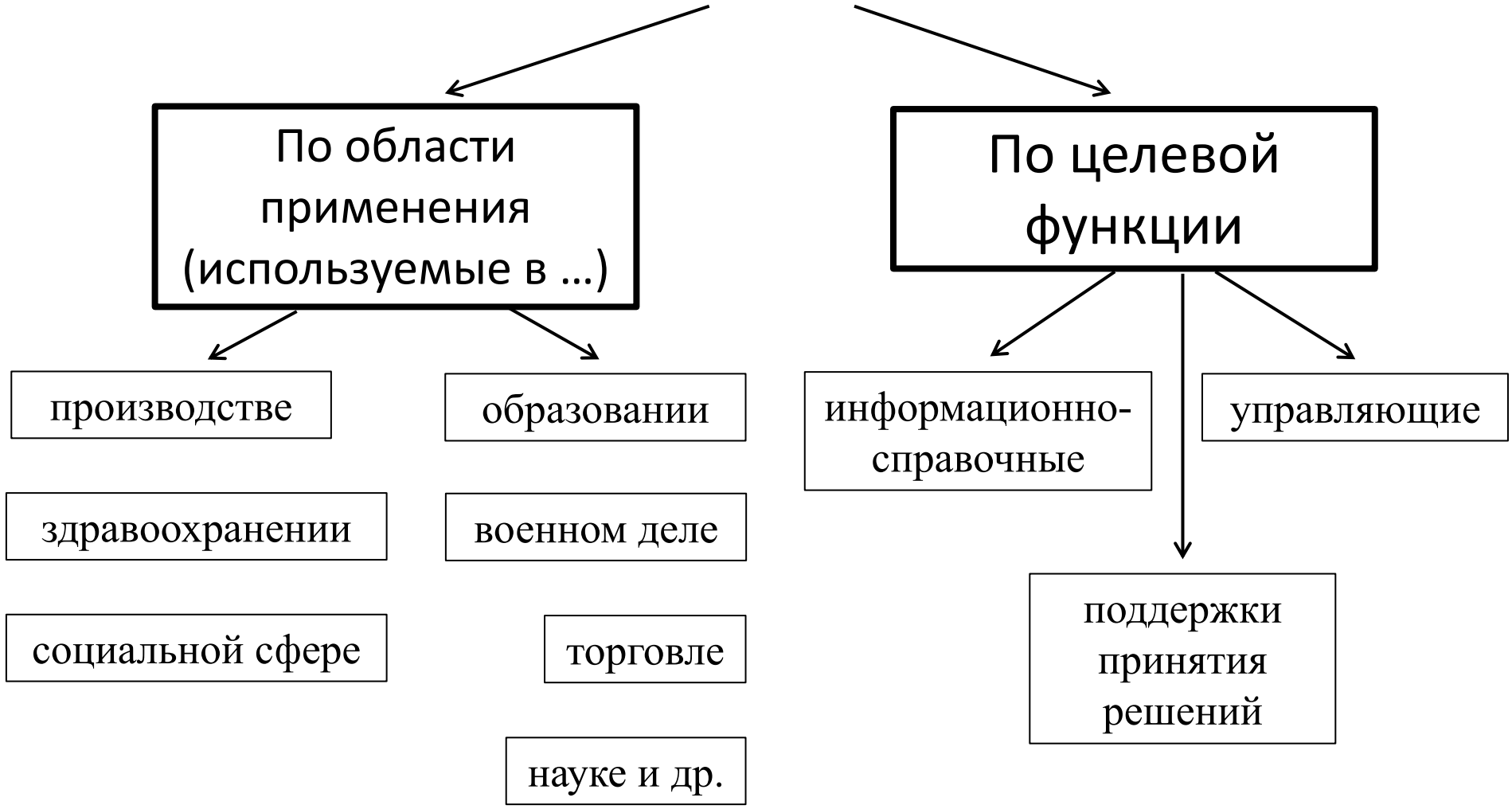
Иногда применяется более узкая трактовка понятия ИС как совокупности аппаратно-программных средств, используемых для решения некоторой прикладной задачи.

Например:

- учет кадров,
- учет материально-технических средств,
- расчет с поставщиками и заказчиками,
- бухгалтерский учет и т. п.

Автоматизированными называют ИС, в которых применяют технические средства, в частности компьютеры. Большинство существующих ИС являются автоматизированными.

Информационные системы



Конкретные задачи, которые должны решаться информационной системой, зависят от той прикладной области, для которой предназначена система.

Но можно выделить некоторое количество задач, не зависящих от специфики прикладной области, а связанных с общими чертами информационных систем.

1) **Обеспечение надежных способов хранения информации**
2) **Хранение данных, обладающих разными структурами**

Требования к ИС:

- Система должна работать с разными файлами данных
- Развитие ИС, появление новых функций, для выполнения которых требуются дополнительные данные с новой структурой
- Вся накопленная ранее информация должна остаться сохранной

3) Обеспечение поддержки соответствия хранимой информации состоянию предметной области

Необходимо предусмотреть дополнительные функции ИС для:

- добавления данных,
- обновления данных,
- удаления данных.

4) Автоматическое обеспечение согласованности действий

- Возможность работы с системой с нескольких рабочих мест
- Обеспечить достоверность и непротиворечивость всех результатов, получаемых от ИС

Целостное состояние БД ИС — это состояние, которое соответствует требованиям прикладной области (или, точнее, требованиям модели прикладной области, на основе которой проектировалась информационная система).

Классическая транзакция — последовательность операций изменения БД и/или выборки из БД, воспринимаемая СУБД как атомарное действие.

- Обеспечение высокой производительности при параллельном выполнении нескольких транзакций

Условие: конечный результат выполнения всего набора транзакций будет эквивалентен результату их некоторого последовательного выполнения

5) Обеспечение распределенной обработки данных

Для решения данной задачи возможно применение двух технологий:

- *технология распределенных БД*, когда различные части БД физически хранятся на различных компьютерах
- *технология тиражирования*, т.е. использование копий одной и той же БД на нескольких компьютерах

6) Обеспечение удобного и соответствующего целям информационной системы пользовательского интерфейса

Таблица поставщиков (s):

n_post	name	rating	town
S1	Smith	20	London
S2	Jones	10	Paris
S3	Blake	30	Paris
S4	Clark	20	London
S5	Adams	30	Athens

Таблица товаров (p):

n_det	name	cvet	ves	town
P1	Monitor screen	Red	12000	London
P2	Computer mouse	Green	150	Paris
P3	Keyboard	Blue	400	Rome
P4	Keyboard	Red	350	London
P5	Power unit	Blue	1200	Paris
P6	Computer case	Red	7500	London

Таблица поставок (sp):

n_post	n_det	date_post	kol
S1	P1	2021-02-01	300
S1	P2	2021-04-05	200
S1	P3	2021-05-12	400
S1	P4	2021-06-15	200
S1	P5	2021-07-22	100
S1	P6	2021-08-13	100
S2	P1	2021-03-03	300
S2	P2	2021-06-12	400
S3	P2	2021-04-04	200
S4	P2	2021-03-23	200
S4	P4	2021-06-17	300
S4	P5	2021-08-22	400

Группирование

Оператор ***group by*** группирует таблицу, представленную фразой ***from*** в группы таким образом, чтобы в каждой группе все строки имели одно и тоже значение поля, указанного во фразе ***group by***. Далее к каждой группе перекомпанованной таблицы (а не к каждой строке исходной таблицы) применяется фраза ***select***, в результате чего каждое выражение во фразе ***select*** принимает единственное значение для группы.

Пример.

Выдать для каждой поставляемой детали ее номер и общий объем поставок, за исключением поставок поставщика 'S1'.

```
Select n_det, sum(kol) from SP
where n_post <>'S1'
group by n_det
```

n_det	sum(kol)
P1	300
P2	800
P4	300
P5	400

Фраза ***having*** играет ту же роль для групп, что и фраза ***where*** для строк и используется для того, чтобы исключать группы, точно так же, как ***where*** используется для исключения строк. Выражение во фразе ***having*** должно принимать единственное значение для группы.

Пример.

Выдать номера деталей, поставляемых более чем одним поставщиком.

```
Select n_det
from SP
group by n_det
having count(*) > 1
```

n_det
P1
P2
P4
P5

Соединения таблиц

Классическая реляционная алгебра Кодда включает девять реляционных операций, последовательное применение которых позволяет реализовать выборку любых данных. Три из этих операции так или иначе связаны с соединением таблиц:

- операция взятия декартова произведения;
- операция соединения (соответствующая ей в стандарте ANSI операция носит название операции внутреннего соединение);
- операция эквисоединения.

Операция взятия декартова произведение содержит все возможные комбинации конкатенаций кортежей (строк) из соединяемых таблиц.

Операция соединения представляет собой соединение кортежей соединяемых таблиц по указанному условию соединения. Строки, которые не удовлетворяют условиям соединения, отбрасываются.

Операция эквисоединения является частным случаем операции соединение по условию равенства атрибутов.

Внешнее соединение может сохранить строки, для которых не находится соответствия в другой таблице. В этом случае недостающие поля заполняются значениями NULL. Решение о том, войдет ли такая строка в результирующий набор, зависит от того, в какой из соединяемых таблиц отсутствуют данные, и от типа внешнего соединения. Существуют три разновидности внешних соединений:

- Левое внешнее соединение. Всегда содержит как минимум один экземпляр каждой строки из таблицы, указанной слева от ключевого слова JOIN. Отсутствующие поля из правой таблицы заполняются значениями NULL.
- Правое внешнее соединение. Всегда содержит как минимум один экземпляр каждой строки из таблицы, указанной справа от ключевого слова JOIN. Отсутствующие поля из левой таблицы заполняются значениями NULL.
- Полное внешнее соединение. Всегда содержит как минимум один экземпляр каждой строки каждой из соединяемых таблиц. Отсутствующие поля в записях результирующего набора заполняются значениями NULL.

Для построения соединений стандарт ANSI предусматривает следующую конструкцию спецификаторов from и join:

FROM *источник1* [*Natural*] *тип соединения* JOIN *источник2* [*on* условие [,...] | *Using* (поле1 [,...])]

- *Источник1*. Первый из соединяемых наборов данных (имя таблицы или подзапрос).
- *[Natural]*. Два набора данных соединяются по равным значениям одноименных полей. Конструкции *On* и *Using* в этом случае недопустимы.

Тип соединения Возможные виды соединений:

1. [Inner] - внутреннее соединение;
2. Left [Outer] - левое внешнее соединение;
3. Right [Outer] - правое внешнее соединение;
4. Full [Outer] - полное внешнее соединение;
5. Cross – декартово произведение;

- *Источник2*. Второй из соединяемых наборов данных (имя таблицы или подзапрос).
- *On условие [,...]*. Отношение между источниками - критерий, аналогичный тому, который задается в конструкции *Where*.
- *Using (поле1 [,...])*. Одноименные поля источников, по совпадающим значениям которых производится соединение. В отличии от *Natural* позволяет ограничиться некоторыми одноименными полями, тогда как *Natural* производит соединение по всем одноименным полям.

Пример. Простое декартово произведение.

Выдать информацию обо всех возможных парах поставщик - деталь.

```
Select S.*, P.*
from S
Cross Join P
```

Результат:

n_post	name	rating	town	n_det	name	cvet	ves	town
S1	Smith	20	London	P1	Monitor screen	Red	12000	London
S2	Jones	10	Paris	P1	Monitor screen	Red	12000	London
S3	Blake	30	Paris	P1	Monitor screen	Red	12000	London
S4	Clark	20	London	P1	Monitor screen	Red	12000	London
S5	Adams	30	Athens	P1	Monitor screen	Red	12000	London
S1	Smith	20	London	P2	Computer mouse	Green	150	Paris
S2	Jones	10	Paris	P2	Computer mouse	Green	150	Paris
S3	Blake	30	Paris	P2	Computer mouse	Green	150	Paris
S4	Clark	20	London	P2	Computer mouse	Green	150	Paris
S5	Adams	30	Athens	P2	Computer mouse	Green	150	Paris
S1	Smith	20	London	P3	Keyboard	Blue	400	Rome
S2	Jones	10	Paris	P3	Keyboard	Blue	400	Rome
S3	Blake	30	Paris	P3	Keyboard	Blue	400	Rome
S4	Clark	20	London	P3	Keyboard	Blue	400	Rome
S5	Adams	30	Athens	P3	Keyboard	Blue	400	Rome
S1	Smith	20	London	P4	Keyboard	Red	350	London
S2	Jones	10	Paris	P4	Keyboard	Red	350	London
S3	Blake	30	Paris	P4	Keyboard	Red	350	London
S4	Clark	20	London	P4	Keyboard	Red	350	London
S5	Adams	30	Athens	P4	Keyboard	Red	350	London
S1	Smith	20	London	P5	Power unit	Blue	1200	Paris
S2	Jones	10	Paris	P5	Power unit	Blue	1200	Paris
S3	Blake	30	Paris	P5	Power unit	Blue	1200	Paris
S4	Clark	20	London	P5	Power unit	Blue	1200	Paris
S5	Adams	30	Athens	P5	Power unit	Blue	1200	Paris
S1	Smith	20	London	P6	Computer case	Red	7500	London
S2	Jones	10	Paris	P6	Computer case	Red	7500	London
S3	Blake	30	Paris	P6	Computer case	Red	7500	London
S4	Clark	20	London	P6	Computer case	Red	7500	London
S5	Adams	30	Athens	P6	Computer case	Red	7500	London

Замечание: тот же результат может быть получен запросом

Select * from S, P

В отличие от предыдущего запроса этот запрос написан с отклонением от стандарта ANSI, но он более точно отражает смысл операции взятия декартова произведения.

При написании запросов следует придерживаться стандарта ANSI, позволяющего формировать более читабельные запросы.

Таблица поставщиков (s):

n_post	name	rating	town
S1	Smith	20	London
S2	Jones	10	Paris
S3	Blake	30	Paris
S4	Clark	20	London
S5	Adams	30	Athens

Таблица товаров (p):

n_det	name	cvet	ves	town
P1	Monitor screen	Red	12000	London
P2	Computer mouse	Green	150	Paris
P3	Keyboard	Blue	400	Rome
P4	Keyboard	Red	350	London
P5	Power unit	Blue	1200	Paris
P6	Computer case	Red	7500	London

Таблица поставок (sp):

n_post	n_det	date_post	kol
S1	P1	2021-02-01	300
S1	P2	2021-04-05	200
S1	P3	2021-05-12	400
S1	P4	2021-06-15	200
S1	P5	2021-07-22	100
S1	P6	2021-08-13	100
S2	P1	2021-03-03	300
S2	P2	2021-06-12	400
S3	P2	2021-04-04	200
S4	P2	2021-03-23	200
S4	P4	2021-06-17	300
S4	P5	2021-08-22	400

Пример. Простое эквисоединение.

Выдать все комбинации информации о поставщиках

и деталях, расположенных в одном городе.

```
Select S.n_post, p.n_det, rating
from S
Cross Join P
where S.town = P.town
```

n_post	n_det	rating
S1	P1	20
S4	P1	20
S2	P2	10
S3	P2	30
S1	P4	20
S4	P4	20
S2	P5	10
S3	P5	30
S1	P6	20
S4	P6	20

```
Select S.*, p.*
from S
Cross Join P
where S.town = P.town
```

n_post	name	rating	town	n_det	name	cvet	ves	town
S1	Smith	20	London	P1	Monitor screen	Red	12000	London
S4	Clark	20	London	P1	Monitor screen	Red	12000	London
S2	Jones	10	Paris	P2	Computer mouse	Green	150	Paris
S3	Blake	30	Paris	P2	Computer mouse	Green	150	Paris
S1	Smith	20	London	P4	Keyboard	Red	350	London
S4	Clark	20	London	P4	Keyboard	Red	350	London
S2	Jones	10	Paris	P5	Power unit	Blue	1200	Paris
S3	Blake	30	Paris	P5	Power unit	Blue	1200	Paris
S1	Smith	20	London	P6	Computer case	Red	7500	London
S4	Clark	20	London	P6	Computer case	Red	7500	London

Замечание: тот же результат может быть получен запросом с использованием конструкции операции внутреннего соединения в стандарте ANSI

```
Select S.n_post, P.n_det, rating
from S
Inner Join P on S.town=P.town
```

n_post	n_det	rating
S1	P1	20
S4	P1	20
S2	P2	10
S3	P2	30
S1	P4	20
S4	P4	20
S2	P5	10
S3	P5	30
S1	P6	20
S4	P6	20

или запросом, написанным с отклонением от стандарта ANSI

```
Select S.n_post, P.n_det, rating
from S, P
where S. town =P. town
```

Пример. Соединение таблиц с дополнительным условием.

Выдать все комбинации информации о поставщиках и деталях, расположенных в одном городе, кроме поставщиков с рейтингом = 20.

```
Select S.n_post, p.n_det, rating
from S
Cross Join P
where S.town = P.town and S.rating < > 20
```

n_post	n_det	rating
S2	P2	10
S3	P2	30
S2	P5	10
S3	P5	30

Пример. Соединение таблицы с ней самой.

Выдать все пары поставщиков из одного города.

```
Select one.n_post, two.n_post
from S one
Cross Join S two
where one.town = two.town  and one.n_post <
two.n_post
```

n_post	n_post
S2	S3
S1	S4

Пример. Внутреннее соединение.

Выдать для каждой поставки номер поставщика,

его фамилию и количество деталей.

```
Select S.n_post, S.name, SP.kol
from SP
inner join S on S.n_post=SP.n_post
```

n_post	name	kol
S1	Smith	300
S1	Smith	200
S1	Smith	400
S1	Smith	200
S1	Smith	100
S1	Smith	100
S2	Jones	300
S2	Jones	400
S3	Blake	200
S4	Clark	200
S4	Clark	300
S4	Clark	400

Пример. Соединение трех таблиц.

Выдать все пары названий городов таких, что какой-либо поставщик, находящийся в первом из этих городов, поставляет деталь, хранимую в другом городе.

```
Select distinct S.town, P.town
from SP
inner join S on S.n_post = SP.n_post
inner join P on P.n_det = SP.n_det
```

town	town
London	London
London	Paris
London	Rome
Paris	London
Paris	Paris

Пример. Простое внешнее соединение двух таблиц.

Пример левого внешнего соединения.

```
Select S.n_post, S.name, SP.kol
from S
left outer join SP on (S.n_post=SP.n_post)
```

n_post	name	kol
S1	Smith	300
S1	Smith	200
S1	Smith	400
S1	Smith	200
S1	Smith	100
S1	Smith	100
S2	Jones	300
S2	Jones	400
S3	Blake	200
S4	Clark	200
S4	Clark	300
S4	Clark	400
S5	Adams	*NULL*

Добавление ключевого слова *outer* перед именем таблицы SP превращает ее в подчиненную таблицу.

Результатом этого внешнего соединения будет получение сведений обо всех поставщиках, независимо от того, делали ли они поставки.

Полное внешнее соединение даст аналогичный результат, правое - результат, аналогичный внутреннему соединению.