



ОНЛАЙН-ОБРАЗОВАНИЕ


Онлайн-образование

Не забыть включить запись!





Меня хорошо видно && слышно?

Ставьте  , если все хорошо
Напишите в чат, если есть проблемы

Подходы к проектированию БД, паттерны

курс “MS SQL Server Developer”



Коробков Виктор

telegram: @Korobkov_Viktor

Правила вебинара



Активно участвуем



Задаем вопрос в чат или голосом



Off-topic обсуждаем в Slack



Вопросы вижу в чате, могу ответить не сразу

Цели вебинара | После занятия вы сможете

1 Производить ER-моделирование базы данных

2 Использовать визуальные средства проектирования

3 Объяснять когда какой подход лучше использовать

Смысл | зачем вам это уметь

Для создания реального рабочего проекта базы данных в любой предметной области

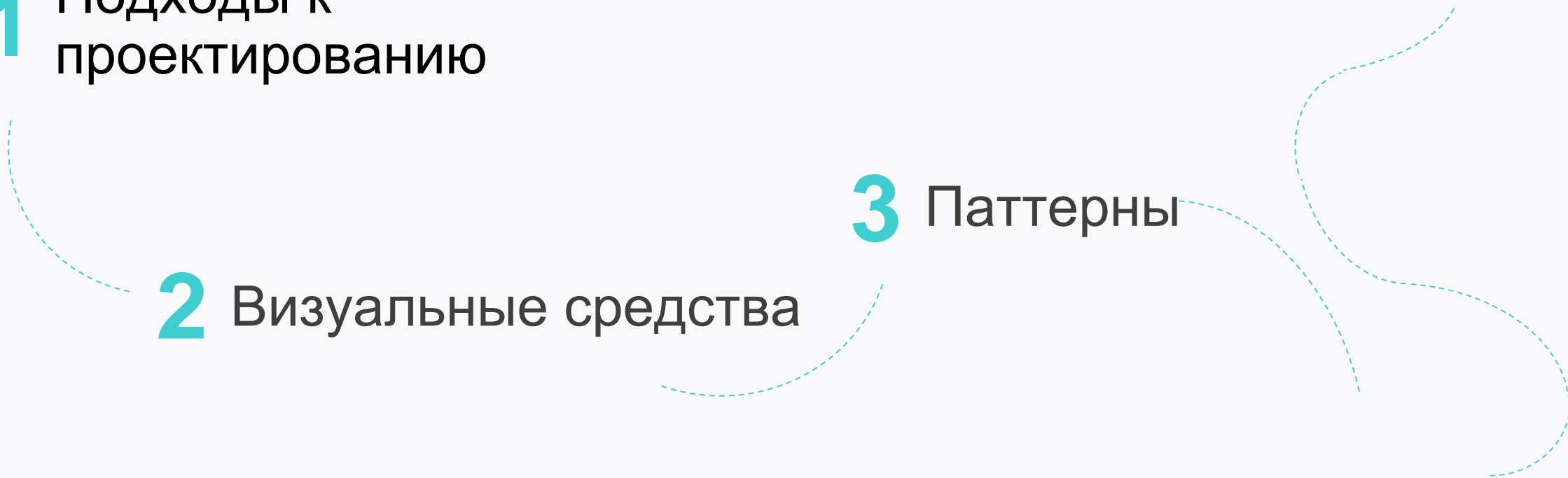
Маршрут вебинара

1 Подходы к проектированию

2 Визуальные средства

3 Паттерны

4 Рефлексия

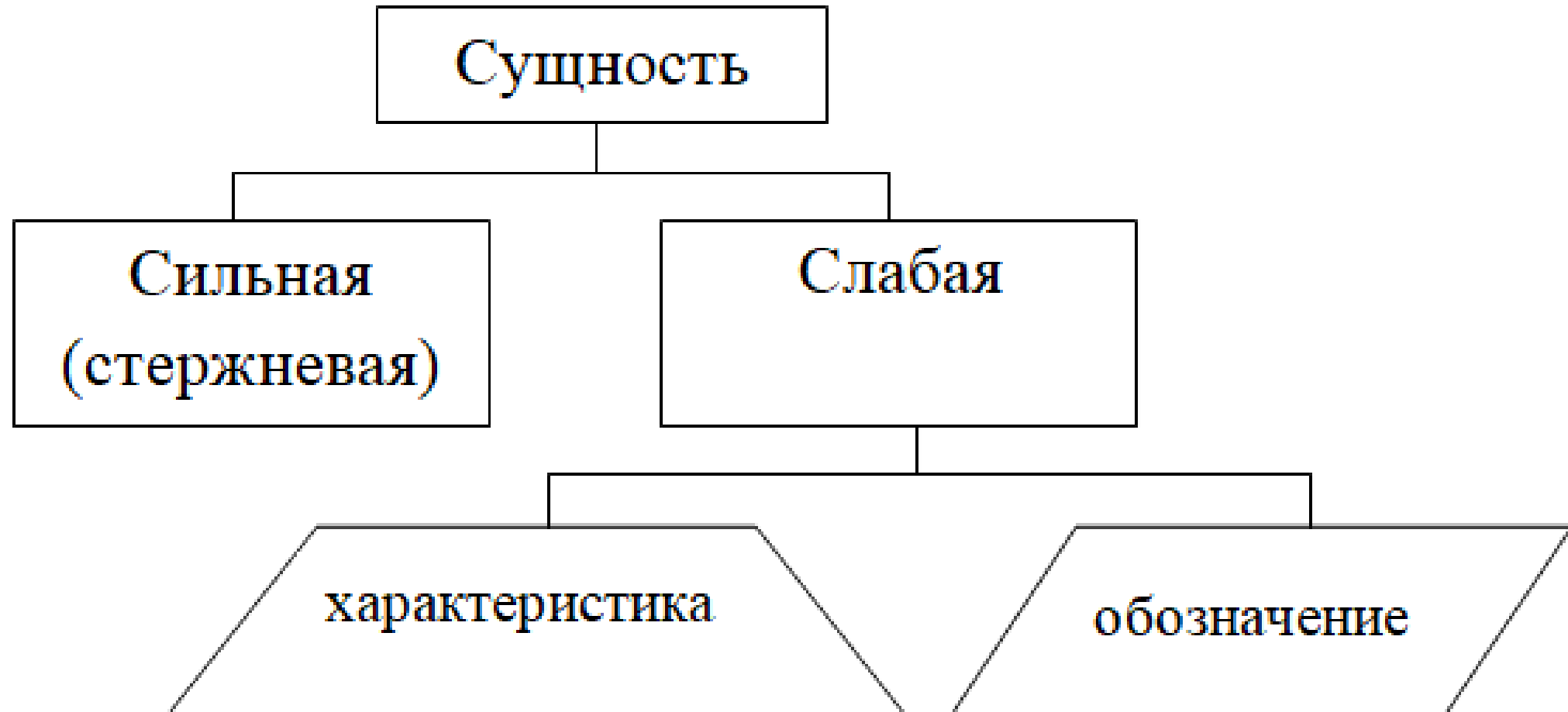


ER - моделирование

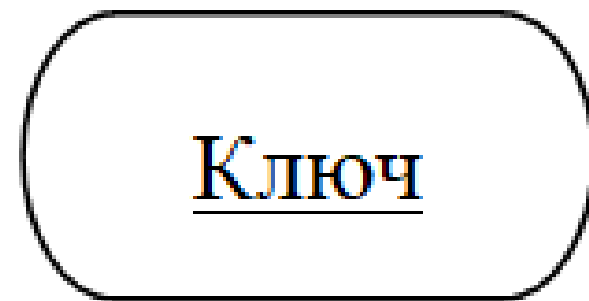
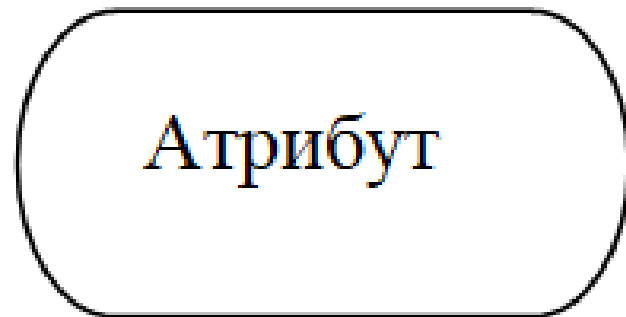
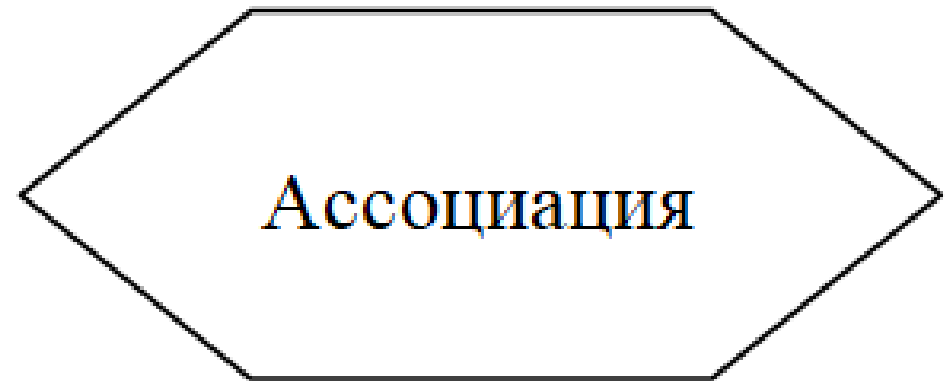
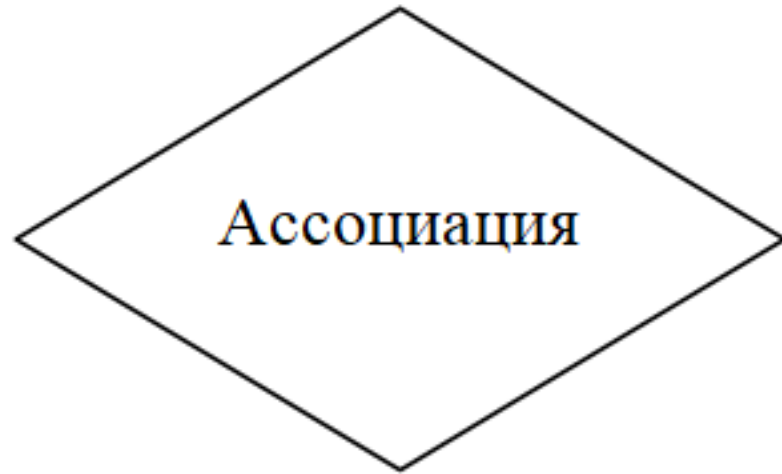
Концепция ER-моделирования:

1. Мир состоит из **объектов**.
2. Объекты образуют **типы**. Каждый объект является **экземпляром** некоторого типа. Объекты одного типа обладают общими **свойствами**.
3. Каждый объект обладает некоторым особым свойством (набором свойств), которое служит для его **идентификации**.
4. Каждый объект может быть связан с другими объектами с помощью **отношений**.

ER - моделирование



ER - моделирование





ER - моделирование

Анализ предметной области:

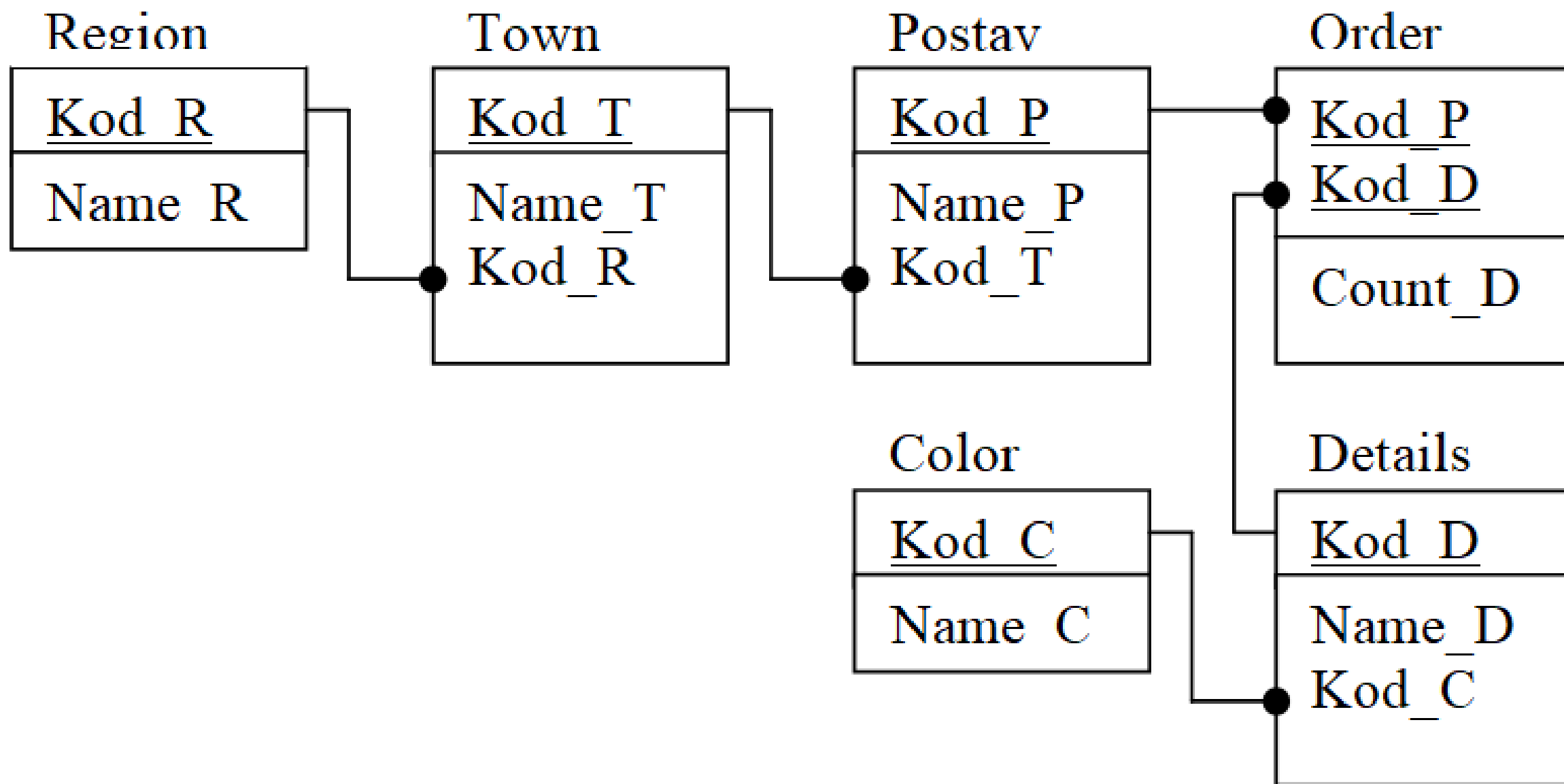
1. Проанализировать ключи сущностей и естественные составные ключи заменить искусственными ключами (кодами).
2. Атрибуты, выбираемые из ограниченного множества значений, представить отдельными сущностями (обозначениями).
3. Выделить из стержневой сущности характеристики связью «один к одному», если:
 - не каждый экземпляр стержневой сущности имеет связанный с ним экземпляр характеристики, то есть это связь не 1:1, а 1:1(0);
 - атрибуты характеристики не слишком часто используются при обработке стержневой сущности.



Концептуальная модель

Правила отображения ER модели на логическую схему

1. Каждая сущность становится отношением, идентификатор сущности становится первичным ключом, а его характеристики - атрибутами отношения.
2. Связь типа «один ко многим» не образует нового отношения, но идентификатор главной сущности становится внешним ключом отношения для дочерней сущности.
3. Связь типа «многие ко многим» становится новым отношением, идентификаторы связываемых сущностей становятся составным первичным ключом отношения для связи, а характеристики (если есть) становятся атрибутами отношения для связи.



Логическая модель

Аспекты проектирования БД

1. Локализация – язык\время.
2. Масштабирование. Насколько быстро будет расти приложение, для которого делается проект?
3. Безопасность и юридические нормы.
4. Вероятность миграции в облако.
5. Политики по хранению данных: будете ли Вы что-то удалять или предпочтете архивирование, история изменений, действий пользователей.
6. Если уже есть какая-то БД, какие основные проблемы с ней?

Визуальные средства проектирования

1. Lucidchart
2. SAP PowerDesigner
3. AllFusion ERwin Data Modeler (ERwin)
4. Visual Paradigm
5. Navicat Data Modeler Essentials
6. Microsoft Visio
7. SQL Server Database Modeler
8. ORACLE SQL Developer Data Modeler
9. MySQL Workbench
10. dbDesigner.net
11. sqldbm.com
12. dbdiagram.io

Генерация документации

1. Red Gate SQL Doc (trial)
2. <http://schemaspy.org> (free)
3. ...

Кейс 1

Name	Type
UserId	Int
UserName	Nvarchar(250)
LastLoginDate	Date
LoginFailCount	Tinyint
DateEmailConfirmed	Datetime2
FirstName	Nvarchar(50)
LastName	Nvarchar(50)
AccountType	Tinyint
...	...

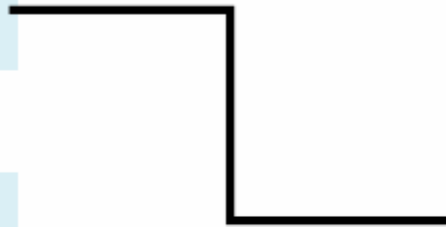
Есть широкая (много полей) таблица. В этой таблице чаще всего используются 7 из 40 полей. Остальные редко читаются и изменяются.

Вы бы хотели иметь эту таблицу поуже, чтобы она занимала меньше места в кэше. Что можно сделать?

Кейс 1

Разбить таблицу на 2 с отношением 1 к 1

Name	Type
UserId	Int
UserName	Nvarchar(250)
LastLoginDate	Date
LoginFailCount	Tinyint

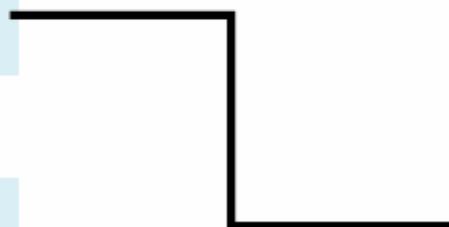


Name	Type
UserId	Int
DateEmailConfirmed	Datetime2
FirstName	Nvarchar(50)
LastName	Nvarchar(50)
AccountType	Tinyint

Персональные данные

Для поддержки требований по хранению персональных данных их лучше вынести в отдельную таблицу.

Name	Type
UserId	Int
UserName	Nvarchar(250)
Login	Nvarchar(250)
SignUpDate	Date



Name	Type
UserId	Int
Email	Nvarchar(150)
FirstName	Nvarchar(50)
LastName	Nvarchar(50)
Phone	Nchar(10)

Кейс 2

Таблицы с часто меняющейся структурой.

У вас есть таблица, например банкомат и его метрики работы и оборудования. Как бы вы сделали структуру?

Или это может быть:

- разные характеристики товаров для разных категорий;
- заявки с разными полями;
- ...

Кейс 2

Вариант 1 - все в поля.

Name	Type
AtmId	Bigint
AtmLocation	Int
CashType	Int
CashStatus	Int
CashErrorCode	NVARCHAR(50)
CashErrorMessage	NVARCHAR(500)
ModemType	Int
ModemStatus	Int
PrinterType	Int
PrinterStatus	Int

Плюсы

Удобно для выборок

Минусы

Каждый раз добавлять новые поля

Кейс 2

Вариант 2 - формирование таблицы с параметрами.

Name	Type
ParamId	Int
ParamDescription	NVARCHAR(500)

Name	Type
AtmId	Bigint
ParamId	Int
ParamValue	NVARCHAR(500)

Name	Type
AtmId	Bigint
ParamId	Int
ParamValueInt	Int
ParamValueText	NVARCHAR(500)
ParamValueDecimal	Decimal(19,4)
ParamValueDatetime	Datetime2

Плюсы: не нужно
менять структуру
таблицы.

Минусы: усложняется
выборка, нужно
заводить новые
параметры в таблицу-
описание.

Кейс 2

Вариант 3 - хранить в xml или json.

Name	Type
AtmId	Bigint
AtmInfo	Json

Плюсы:

не нужно менять структуру;
ничего не нужно делать при
добавлении новых данных;
удобство выборки.

Минусы:

уровень свободы структуры.

Кейс 2

Вариант 4. Oldschool

Антипаттерн

НЕ Храните данные в строке через разделитель

Name	Type
AtmId	Bigint
AtmInfo	Nvarchar(MAX)

Плюсы

Не нужно менять структуру
Вообще ничего не нужно делать при добавлении новых данных

Минусы

Уровень свободы структуры
Уровень ошибок в данных
Неудобно делать выборки на SQL

Иерархия в БД

Вариант 1.

Список смежности (adjacency list) - хранение parent_id

Employee	Manager (Parent)
Пупкин	NULL
Иванов	Пупкин
Петров	Иванов
Сидоров	Иванов

Вариант 2.

Соединительная таблица (bridge table)

Employee	Manager (Parent)	NodeLevel	LevelFrom Parent
Пупкин	NULL	0	0
Иванов	Пупкин	1	1
Петров	Иванов	2	1
Петров	Пупкин	2	2
Сидоров	Иванов	2	1
Сидоров	Пупкин	2	2

Изменение схемы данных

Обратимые изменения:

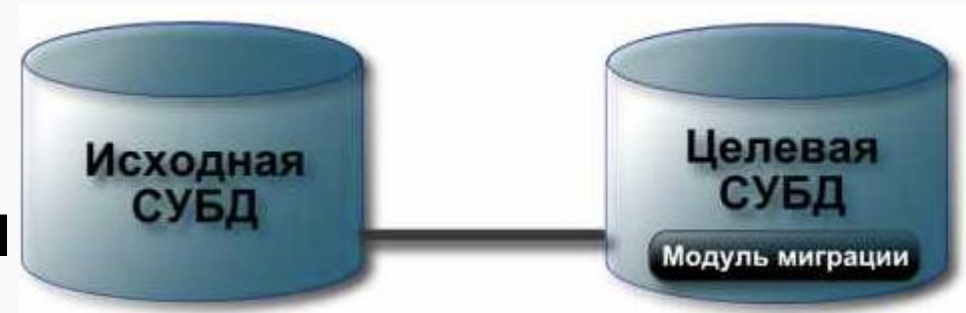
- добавления поля;
- создание таблицы;
- удаление процедур/функций.

Необратимые/сложно обратимые изменения:

- удаление столбца/таблицы;
- удаление/изменение данных из таблицы;
- изменение типа.

Изменение схемы данных

1. Вручную
2. Скриптами, которые написаны вручную
3. Сгенерированными скриптами
4. Специальным ПО



Кейс № 3

1. У вас БД с работающим приложением, всего 1 сервер (монолит)
2. Пользователи активны только в бизнес часы
3. На проекте 3 разработчика, 1 ДБА
4. Нужно создать дополнительные поля в небольшой таблице и поправить процедуры
5. Как бы вы делали миграцию? Способ, и описание этапов (что за чем)

Кейс № 4

1. У вас БД с работающим приложением, 10 серверов БД с одинаковым кодом в БД, но разными данными (шардинг).
2. В основном пользователи наиболее активны в бизнес часы, но продолжают работать и после работы и в выходные
3. На проекте 5 разработчика, 2 ДБА
4. Нужно создать 4 дополнительных поля в небольшой таблице и поправить 3 процедуры
5. Как бы вы делали миграцию? Способ, и описание этапов.

Обобщение по изменению схемы

Метод	Количество серверов	Активность пользователей	Объем данных в таблице
Вручную	1-2	эпизодическая	до 100 млн
Написанные скрипты	> 1	постоянная	любой
Скрипты сгенерированные ПО	> 1	постоянная	любой
Специальное ПО	любое	эпизодическая	до 100 млн

Добавление поля со значением по умолчанию

Postgres, MySQL, Oracle – добавит значение в любом случае
SQL Server – добавит поле и будет вписывать значение для новых строк только если есть NOT NULL

1. Добавляем поле с Null.
2. Запускаем фоновый скрипт, который меняет по кусочкам это поле на нужное значение.
3. Прописываем дефолт на таблице.

Добавление (удаление) индекса

1. Добавить новый индекс.
2. Удалить старый индекс.

Только в таком порядке !!!



Флаги версии (feature flag)

Флаги включения или выключения новых фич.

Флаги можно хранить:

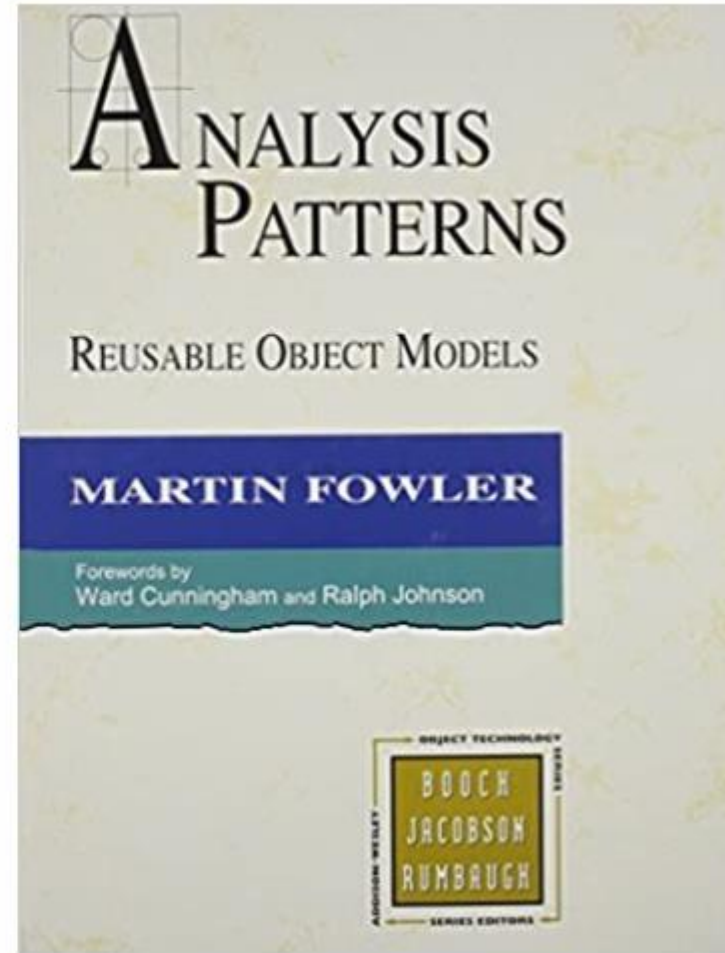
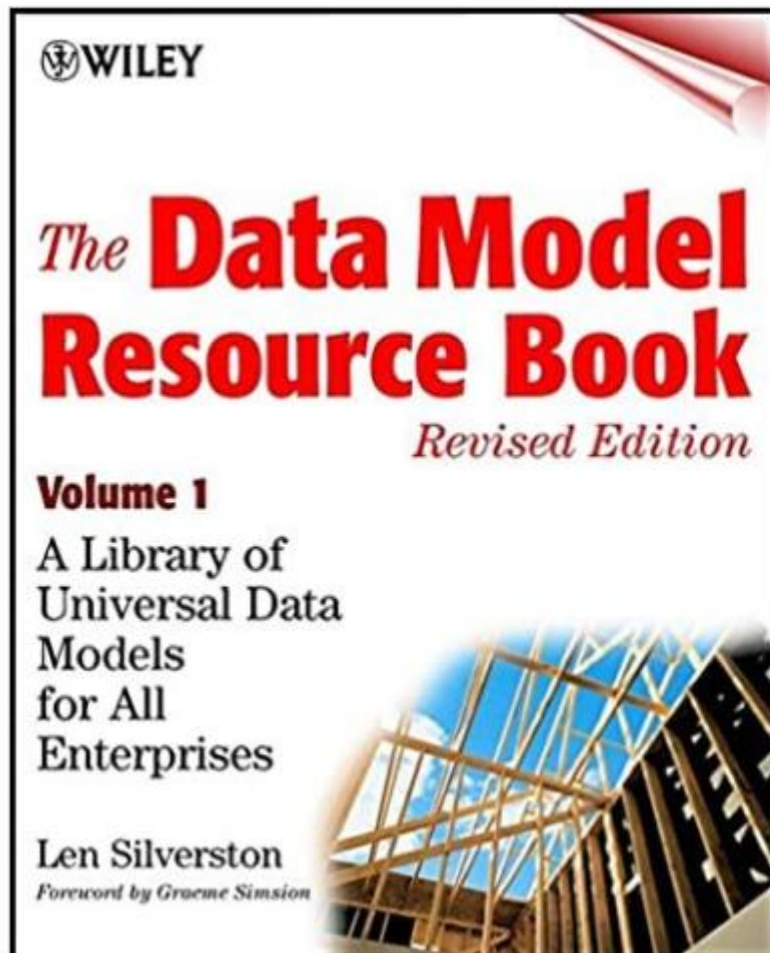
- в таблице;
- сделать функцию, которая будет возвращать 1 или 0.

В процедуре код работает в зависимости, от значения флага. Код переносится на прод, флаг включается, если что выключается.

Делаете флаги в зависимости от группы пользователей/компаний.

Дополнительные материалы

Примеры описания различных предметных областей:
http://www.databaseanswers.org/data_models/index.htm



Рефлексия

О чем мы сегодня говорили?

- Что такое ER модель, когда ее применяют?
- Зачем делают таблицы со связью 1 к 1?

The background of the entire image is an aerial photograph of a city with many skyscrapers, overlaid with a semi-transparent blue layer. A network of white lines connects various points across the blue area, creating a digital or technological aesthetic.

Заполните, пожалуйста,
опрос о занятии по ссылке в чате

Спасибо за внимание!
Приходите на следующие вебинары



Коробков Виктор