Базы данных

Базу данных можно определить как совокупность взаимосвязанных хранящихся вместе данных при наличии такой минимальной избыточности, которая допускает их использование оптимальным образом для одного или нескольких приложений.

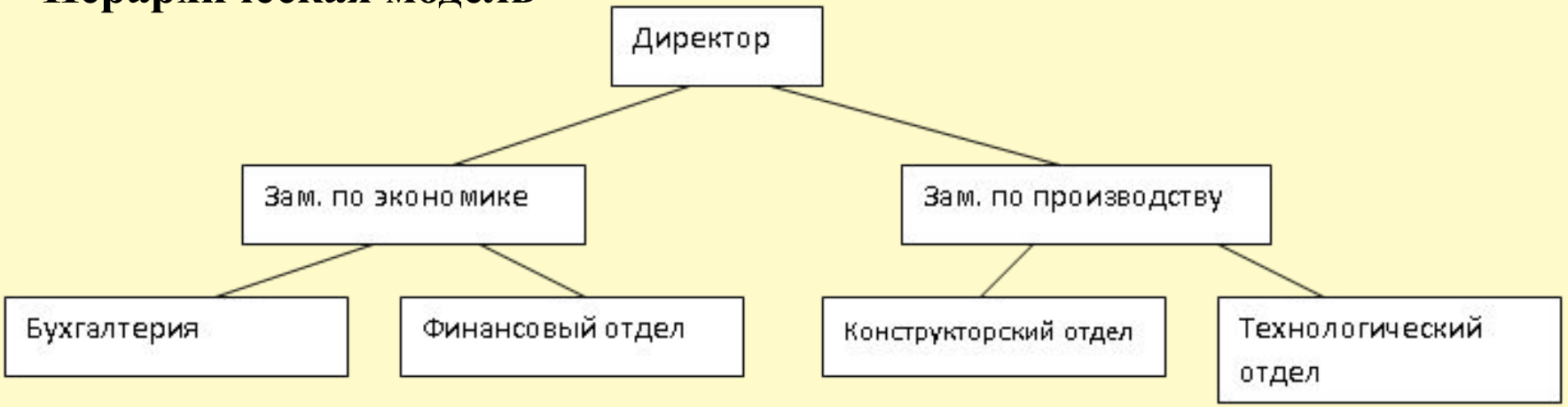
База данных – это совокупность сведений об объектах, процессах, событиях или явлениях, устроенная таким образом, чтобы обеспечить удобное представление этой совокупности как вцелом, так и любой её части.

Появление понятия "База данных" обусловлено возникновением нового класса невычислительных задач, при решении которых используются общие данные.

В качестве основного критерия оптимальности функционирования базы данных, как правило, используются временны́е характеристики реализации запросов пользователей прикладными программами.

Модели данных

Иерархическая модель



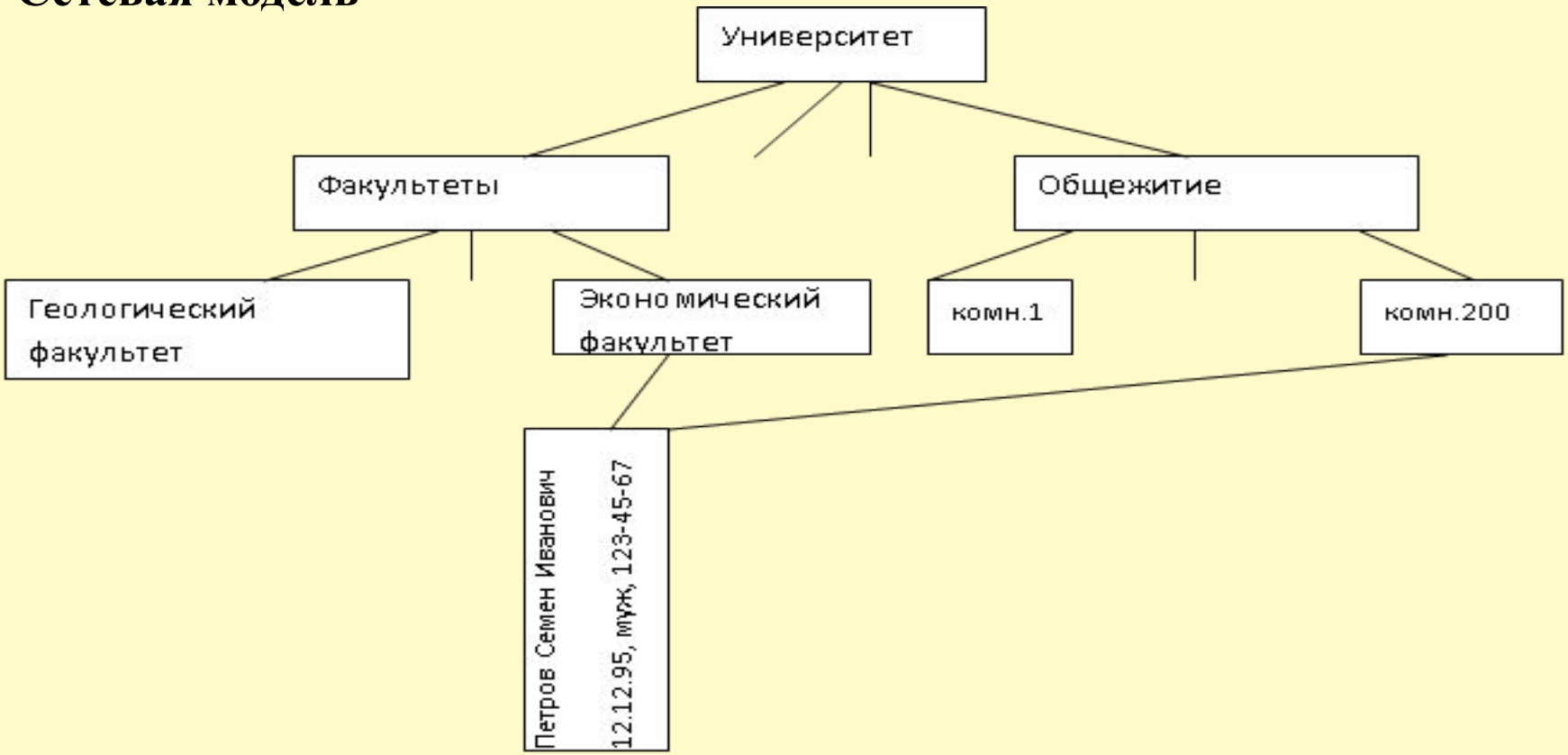
К каждому элементу данных есть только один маршрут по дереву, никакие горизонтальные связи между элементами не допускаются, то есть связи могут быть только между потомками и предками (вертикальные), а не между ветвями.

Достоинство - высокая скорость поиска данных, если известен путь к ним.

Недостаток такой организации связей – необходимость дублирования некоторых данных на разных ветвях.

Модели данных

Сетевая модель



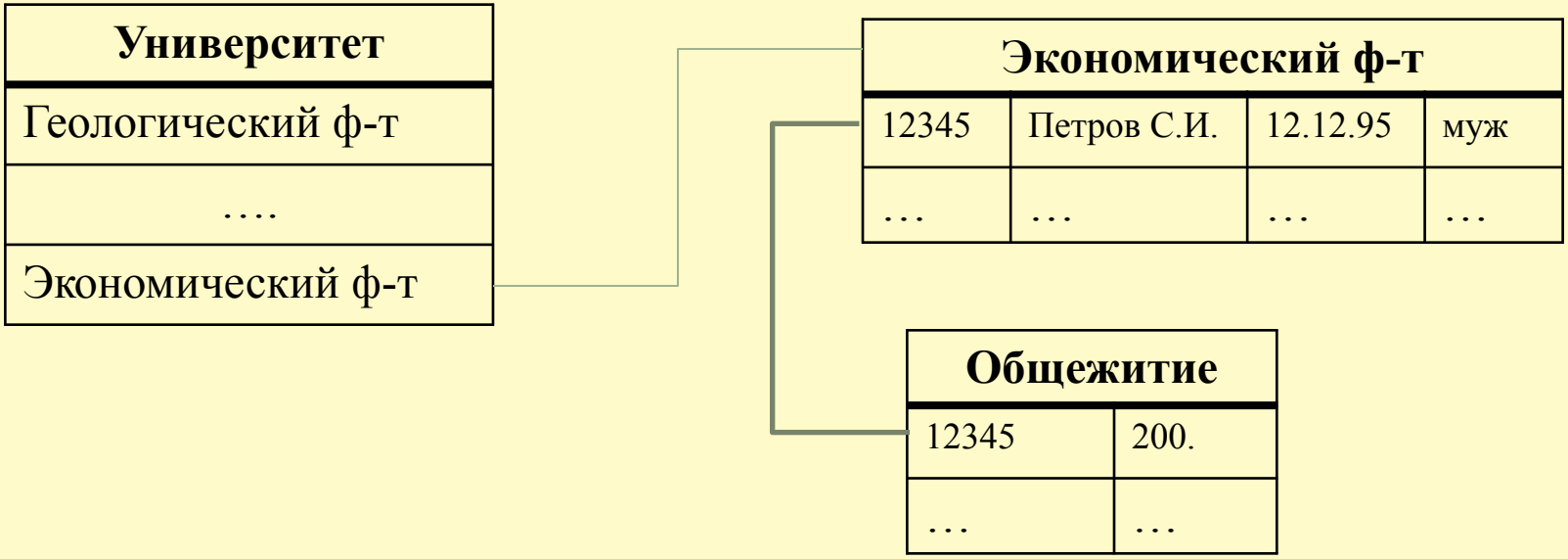
Допускаются как вертикальные, так и горизонтальные связи между элементами данных.

Достоинство – меньшая избыточность данных

Недостаток – сложность разработки.

Модели данных

Реляционная модель



База данных в реляционной модели – это совокупность взаимосвязанных таблиц с информацией.

В каждой таблице хранятся данные об объектах определенного вида.

Все таблицы связаны между собой.

Основные понятия реляционной модели

Запись (строка таблицы) – совокупность сведений о каком-то одном объекте.

Поле (столбец таблицы) – какой-то признак или характеристика объекта.

Каждое поле имеет строго определенный тип и размер.

Ключ – это поле или совокупность нескольких полей таблицы, однозначно определяющих запись.

В таблице не может быть двух записей с одинаковым значением ключа.

Ключи используются для:

- предотвращения дублирования записей,
- для связывания таблиц
- для быстрого поиска нужной записи по её ключу.

Индекс – это средство быстрого поиска данных в таблицах с большим количеством записей. При необходимости индексов в таблице может быть несколько.

Основные понятия реляционной модели

При назначении ключа следует учитывать следующее:

- ключевое поле никогда не может быть пустым
- в составе ключа используется минимально возможный набор полей
- из двух возможных ключей выбирают обычно более короткий
- введенные значения ключевых полей в таблице по возможности должны как можно реже изменяться

Выбор того или иного поля в качестве ключевого зависит от назначения данных.

ФИО	пол	телефон	номер паспорта	номер зачетки	номер читательского билета	адрес

Если это сведения о студентах ВУЗа, то целесообразно в качестве ключевого выбрать поле номер зачетки.

Если это сведения о читателях городской библиотеки, то ключевым лучше сделать номер читательского билета.

Назначение ключей

Для правильного назначения ключа нужно знать некоторые особенности заполнения таблиц.

Пример.

Имеется таблица Экзамены, в которой должны храниться сведения о сдаче студентами экзаменов за всё время обучения. Требуется назначить ключ.

номер зачетки	дата сдачи	предмет	оценка	экзаменатор
123456	02.06.11	история	5	Иванов
123456	04.06.11	химия	5	Петров
123456	09.06.11	физика	4	Семенов
123456	15.06.11	математика	4	Васильев
123458	02.06.11	история	5	Иванов
123458	04.06.11	химия	3	Петров
123458	09.06.11	физика	3	Семенов
123458	15.06.11	математика	3	Васильев
…	…	…	…	…

Назначение ключей

Если в учебном заведении действует железное правило, что *в течение дня студент ни при каких обстоятельствах не может сдать более одного экзамена*, то ключом можно назначить ***номер зачетки +дата сдачи***. Если хотя бы иногда это правило может нарушаться, то такой ключ не годится, так как некоторые записи (с указанным нарушением) будет просто невозможно ввести в таблицу.

Если *названия предметов, которые сдает студент за всё время обучения, никогда не повторяются*, то ключом может быть ***номер зачетки + предмет***. При этом названия одних и тех же предметов, сдаваемых в разных семестрах, обязательно должны отличаться, например, математика-1, математика-2 и т.п.

Если ни одно из вышеназванных правил не выполняется, то ключом можно сделать ***номер зачетки +дата сдачи + предмет***.

Это будет означать, что *в течение одного дня один и тот же студент по указанному предмету может получить только одну оценку*.

Таким образом, при назначении ключей в таблицах нужно прогнозировать процесс заполнения таблиц данными.

Система управления базами данных

Система управления базами данных (СУБД) — совокупность программных и лингвистических средств общего или специального назначения, обеспечивающих управление созданием и использованием баз данных.

Основные функции СУБД:

- Определение структуры создаваемой базы данных
- Предоставление пользователям возможности манипулирования данными
- Обеспечение независимости прикладных программ и данных
- Защита логической целостности базы данных
- Защита физической целостности
- Управление полномочиями пользователей на доступ к базе данных
- Синхронизация работы нескольких пользователей
- Управление ресурсами среды хранения
- Поддержка деятельности системного персонала

Системы управления базами данных

- dBase III
- FoxPro
- Paradox
- Access

настольные СУБД

- MySQL
- MS SQL Server
- Oracle

клиент-серверные СУБД

Системы управления базами данных

Настольные СУБД

Преимущества:

- наличие графического интерфейса пользователя
- простота создания баз данных и пользовательских приложений
- низкие затраты на создание приложений и обслуживание

Недостатки:

- многопользовательский режим работы возможен, но не является основным
- обработка данных выполняется внутри пользовательского приложения и может оказаться длительной по времени
- большой объем трафика при многопользовательском режиме
- отсутствие ограничения доступа к данным на уровне полей
- рост объёмов хранимой информации существенно снижает производительность и надёжность
- отсутствие механизма транзакций

Системы управления базами данных

Клиент-серверные СУБД

Функции приложения-клиента:

- Посылка запросов серверу.
- Интерпретация результатов запросов, полученных от сервера.
- Представление результатов пользователю в некоторой форме (интерфейс пользователя).

Функции серверной части:

- Прием запросов от приложений-клиентов.
- Интерпретация запросов.
- Оптимизация и выполнение запросов к БД.
- Отправка результатов приложению-клиенту.
- Обеспечение системы безопасности и разграничение доступа.
- Управление целостностью БД.
- Реализация стабильности многопользовательского режима работы.

Системы управления базами данных

Клиент-серверные СУБД

Преимущества:

- рассчитаны на многопользовательский режим работы
- обработка запросов выполняется оптимальным образом на сервере, что повышает производительность
- снижение сетевого трафика
- гибкая система управления доступом к данным
- поддержка механизма транзакций
- встроенные средства архивации и восстановления данных
- правила целостности данных определяются в базе данных на сервере и являются едиными для всех приложений

Недостатки:

- требуется разработка клиентских приложений
- нужен администратор баз данных

Этапы разработки базы данных

1. **Постановка задачи.**

Определение целей разработки, основных требований к системе, примерных сроков, и стоимости работ :

- сколько примерно человек должны пользоваться базой
- примерные объемы информации
- как часто появляются и изменяются данные
- будет ли система развиваться в дальнейшем
- должна ли она быть автономной или являться частью другой информационной системы
- каковы требования к защите информации от посторонних
- насколько серьезной должна быть защита от сбоев
- каковы требования по скорости доступа к информации
- какого рода информация должна храниться
- и т.д.

Этапы разработки базы данных

2. Разработка информационно-логической (инфологической) модели

- детальное обследование предметной области
- определение перечня входной и выходной информации
- фиксация детальных характеристик информации
- выявление связей между отдельными объектами предметной области

Результат разработки нередко представляется в виде ER-диаграммы ("сущность-связь«).

При этом еще не решаются технические вопросы выбора оборудования, СУБД и т.п.

Этапы разработки базы данных

3. Выбор СУБД. Разработка логической модели базы данных

- принимается решение об используемой СУБД (на основании анализа выполнения этапов 1 и 2)
- на основе инфологической модели создается детальное описание данных в терминах выбранной СУБД (**логическая модель**)
- производится распределение данных по таблицам, описывается структура каждой таблицы (состав и характеристики полей, ключи, индексы, связи и т.п.)
- производится уточнение инфологической модели

Если выбранная СУБД по каким-то параметрам не подходит., в этом случае производится или изменение требований к системе или выбирается другая СУБД.

Этапы разработки базы данных

4. Разработка программного обеспечения базы данных

- таблицы заполняются данными контрольного примера
- разрабатываются дополнительные объекты базы данных (запросы, формы, отчеты, модули и т.п.)
- результаты разработки проверяются на контрольном примере
- производится согласование результаты с персоналом, который в будущем будет работать с базой
- составляются инструкции как для будущих администраторов базы, так и для пользователей

Этапы разработки базы данных

5. Заполнение базы рабочими данными и поддержание ее в актуальном состоянии

- Производится первичное обучение пользователей
- Вводятся необходимые для дальнейшей работы данные
- Разрабатываются и внедряются организационные документы, закрепляющие обязанности персонала при работе с базой
- Выполняются необходимые доработки по вопросам, выявившимся в процессе эксплуатации

Приведенные этапы характерны для достаточно больших проектов. Для более мелких задач некоторые этапы могут объединяться или отсутствовать. Например, если СУБД заранее известна, то выбирать ее не нужно, но проверить, насколько ее возможностей достаточно для поставленной задачи, стоит. Иначе разработка может зайти в тупик или созданная база данных по своим характеристикам окажется далекой от совершенства.

Объекты базы данных MS Access

- **Таблицы –** основное хранилище информации базы данных
- **Запросы –** выборки данных из таблиц в соответствии с некоторыми условиями
- **Формы –** средства наглядного представления данных на экране
- **Отчеты –** документы, предназначенные для печати
- **Макросы –** минипрограммы на языке макрокоманд для автоматизации некоторых действий с базой данных
- **Модули –** программы на языке VBA (Visual Basic for Applications) для обработки данных
- Страницы доступа к данным (в последних версиях не поддерживаются) – Web-страницы для работы с базой данных

Типы данных Access

- Текстовый – для хранения любых символов. (допустимая длина 1…255 символов)
- Числовой

Целые числа
- Байт(0…255)
- Целое (-32768…32767)
- Длинное целое (≈ -2,1…2,1млрд.)

С плавающей запятой
- Одинарной точности
- Двойной точности

- Денежный – для хранения денежных величин. Отсутствует ошибка округления
- Дата/время – для хранения даты и(или) времени
- Логический – для хранения данных типа Да/Нет

Типы данных Access

- **Поле мемо** – для хранения символьных (текстовых) данных, длина которых непредсказуема и может превышать 255 символов
- **Гиперссылка** – для хранения не самой информации, а ее адреса. Адрес может быть как локальным, так и глобальным
- **Объект OLE** (Вложение) – для хранения документов, созданных в других приложениях (документов Word, таблиц Excel, изображений, аудио – и видеозаписей и т. п.)
- **Счетчик** – автоматически заполняемое поле типа **длинное целое**. Гарантируется неповторяемость данных в этом поле.

Способы создания таблиц в Access

- **С помощью мастера –** предлагается выбрать шаблон таблицы по ее назначению и указать, какие поля из готового списка включать.

  Структура таблицы обычно требует дополнительной правки

- **Посредством ввода данных -** типы и размеры полей назначаются автоматически по первым введенным записям.

  Структура таблицы обычно требует дополнительной правки

- **Режим конструктора –** структура таблицы полностью определяется пользователем

Правила назначения имен полей в Access

- Имя поля не может начинаться с цифры или пробела
- В именах разрешается использовать только буквы, цифры (начиная со второго символа имени), пробелы и символы подчеркивания
- Имена полей в пределах одной таблицы не могут повторяться
- Длина имени поля не более 64 символов
- Заглавные и строчные буквы не различаются
- Имена не должны совпадать с именами встроенных функций (SIN, COS, DATE и др.)
- Имя должно быть осмысленным (обычно отражает содержимое поля)
- В Access 2003 избегать пробелов в именах полей, набранных кириллицей

Виды связей между таблицами в Access

Связь «**один к одному**»:

Каждой записи первой таблицы соответствует 0 или 1 запись второй таблицы.

Такая связь целесообразна, если

- Записей во второй таблице гораздо меньше, чем в первой
- Длина записи первой таблицы превышает допустимую (часть полей из первой таблицы переносится во вторую)

В остальных случаях можно недостающие поля из второй таблицы добавить в первую и обойтись без второй таблицы.

Виды связей между таблицами в Access

Связь «**один ко многим**»:

Каждой записи первой таблицы соответствует 0 ,1 или несколько записей второй таблицы.

Это наиболее часто встречающийся вид связей между таблицами.

Связь «**многие ко многим**»:

Каждой записи первой таблицы соответствует 0 ,1 или несколько записей второй таблицы и наоборот.

Напрямую между двумя таблицами в Access такую связь не создать, но можно организовать через промежуточную таблицу.

Нормализация таблиц

Нормализация таблиц используется для устранения потенциальных противоречий при добавлении, изменении и удалении данных.

Таблица находится в **первой нормальной форме (1NF)**, если каждый её атрибут (поле) атомарен, то есть может содержать только одно значение.

Если в поле хранится набор значений, то это не **1NF.**

Атомарность (неделимость) поля определяется, исходя из возможных вариантов использования содержимого поля.

Нормализация таблиц

Таблица находится во **второй нормальной форме (2NF)**, если она находится в первой нормальной форме, и при этом любой её атрибут (поле), не входящий в состав первичного ключа, функционально полно зависит от первичного ключа.

Функционально полная зависимость означает, что атрибут (поле) функционально зависит от всего первичного составного ключа, но при этом не находится в функциональной зависимости от какой-либо из входящих в него атрибутов(полей).

В 2NF нет неключевых атрибутов, зависящих от части составного ключа.

2NF гарантирует, что каждый атрибут на самом деле является атрибутом этой сущности.

Нормализация таблиц

N зачетки	Фамилия	Экзамен	Оценка
123	Петрова	История	5
123	Петрова	Математика	4

Ключ: № зачетки + Экзамен

Это 1NF, но не 2NF, т.к. Фамилия не зависит от экзамена.

Для приведения к 2NF таблицу следует разбить на две:

N зачетки	Фамилия
123	Петрова

N зачетки	Экзамен	Оценка
123	История	5
123	Математика	4

Нормализация таблиц

Таблица находится в **третьей нормальной форме** (3NF), если она находится во второй нормальной форме (2NF) и при этом любой ее неключевой атрибут зависит *только* от первичного ключа, то есть отсутствуют транзитивные зависимости неключевых атрибутов от ключевых.

Зависимость называется **транзитивной**, если один атрибут (поле), зависит от другого, а тот зависит от первичного ключа.

Обычно для решения практических задач приведения таблиц к 3NF достаточно.

Нормализация таблиц

Пример.

Фамилия	Должность	Оклад
Иванов	Кладовщик	15000
Петрова	Секретарь	20000

Ключ: Фамилия

Если оклад определяется штатным расписанием, то зависит от должности, но не от фамилии. Это не 3NF.

Для приведения к 3NF таблицу следует разбить на две:

Фамилия	Должность
Иванов	Кладовщик
Петрова	Секретарь

Должность	Оклад
Кладовщик	15000
Секретарь	20000