ПРОЕКТИРОВАНИЕ БАЗЫ ДАННЫХ «ИЗДАТЕЛЬСТВО»

Системный анализ предметной области

База данных создаётся для информационного обслуживания редакторов, менеджеров и других сотрудников компании.

БД должна содержать:

1. данные о сотрудниках компании
2. данные о книгах
3. данные об авторах
4. финансовое состояние компании
5. возможность получать разнообразные отчёты.

В соответствии с предметной областью система строится с учётом следующих особенностей:

а) каждая книга издаётся в рамках контракта;

б) книга может быть написана несколькими авторами(не более 5);

в) контракт подписывается одним менеджером и всеми авторами книги;

г) каждый автор может написать несколько книг (по разным контрактам);

д) порядок, в котором авторы указаны на обложке, влияет на размер гонорара;

е) если сотрудник является редактором, то он может работать одновременно над несколькими книгами;

ж) у каждой книги может быть несколько редакторов, один из них – ответственный редактор;

з) каждый заказ оформляется на одного заказчика;

и) в заказе на покупку может быть перечислено несколько книг.

Выделим базовые сущности этой предметной области:

Сотрудники компании. Атрибуты сотрудников:

- ФИО
- табельный номер
- пол
- дата рождения
- паспортные данные
- ИНН
- должность
- оклад
- адрес сотрудника
- телефоны

Для редакторов необходимо хранить сведения о редактируемых книгах;

Для менеджеров – сведения о подписанных контрактах;

Для авторов - сведения о написанных книгах.

Книги. Атрибуты книги:

- авторы
- название
- категория
- количество страниц
- аннотация
- тираж
- дата выхода
- цена одного экземпляра
- общие затраты на издание
- авторский гонорар
- авторский знак

Контракты будем рассматривать как связь между авторами, книгами и менеджерами. Атрибуты контракта:

- Номер
- дата подписания
- участники

Авторы. Атрибуты авторов :

- ФИО автора
- паспортные данные
- адрес автора
- телефоны
- ИНН

Для отражения финансового положения компании в системе нужно учитывать **Заказы** на книги. Атрибуты заказа:

- номер заказа
- дату поступления заказа
- дату его выполнения
- список заказанных книг с указанием количества экземпляров
- стоимость заказа

Заказчики. Атрибуты:

- номер заказчика
- адрес заказчика
- телефоны
- номер заказа

Функции менеджера:

1. Принятие заказов
2. Утверждение плана работ

3. Подписание контрактов
4. Формирование базы данных о заказчиках
5. Контроль оплаты заказов
6. Координация работы сотрудников
7. Составление отчетов

Функции ответственного редактора:

1. Поиск авторов
2. Формирование базы данных об авторах
3. Планирование работ
4. Отбор и редактирование материалов
5. Оформление книги
6. Контроль своевременности сдачи редакционных материалов

Функции автора:

1. Поиск материалов
2. Своевременная сдача материалов
3. Подписание контракта

Функции заказчика:

1. Оформление заказа
2. Оплата заказа

Выводы.Система должна обеспечивать:

а) Поиск книги по кодам, авторам, названиям
б) Поиск книг авторов
в)Поиск книг, которые редактировал данный редактор
г)Поиск редакторов, работающих над данной книгой
д)Поиск авторов когда-либо печатавшихся в данном издательстве
е)Поиск заказчиков когда-либо делавших заказ
ж)Поиск заказов по номеру заказчика, номеру заказа, коду контракта
з)Поиск заказов по стоимости
и)Поиск контрактов, подписанных данным менеджером
к) Фиксировать дату принятия и сдачи заказа
л) Отслеживать задержки заказов

В редакции около 150 авторов, постоянно печатающихся не более 50. Срок редактирования каждой книги не более 6 месяцев. Печать новых книг – до 50 наименований, тиражом до 15000 экземпляров в месяц. В редакцию ежемесячно обращается до 1000 заказчиков. В среднем каждый из них заказывает 30 книг тиражом 300 экземпляров. Отпускная цена издательства составляет примерно 50 рублей за одну книгу средним объемом 300 с.

Логическое проектирование базы данных

Логическое проектирование основано на модели логического уровня и представляет собой описание и построение схем связей между элементами данных безотносительно к их содержанию и среде хранения.

Логическая структура БД получается преобразованием концептуальной схемы в логическую схему (модель), ориентированную на выбранную СУБД.

Применительно к наиболее распространенной реляционной модели данных общий подход преобразования концептуальной схемы в логическую состоит в том, что каждую сущность, являющуюся представителем множества однотипных объектов, задают схемой отдельного отношения (таблицы), а атрибуты сущности образуют столбцы таблицы. Первичный ключ сущности образует исходный первичный ключ таблицы, который в дальнейшем может быть изменен.

Проектирование логической структуры БД должно решать задачи выбора наиболее эффективной структуры данных, обеспечения быстрого доступа к данным; исключения дублирования данных, обеспечения целостности данных таким образом, чтобы при изменении одних объектов автоматически происходило соответствующее изменение связанных с ними объектов.

При неправильно спроектированной схеме БД могут возникнуть аномалии модификации данных. Они обусловлены отсутствием средств явного представления типов множественных связей между объектами ПО и неразвитостью средств описания ограничений целостности на уровне модели данных. Для решения подобных проблем проводится нормализация отношений, предложенная Э.Ф. Коддом в рамках реляционной модели данных.

Нормализация

Целью нормализации является исключение избыточности данных, которая является причиной ошибок при вводе и нерационального использования дискового пространства компьютера.

Нормализация представляет собой процесс последовательного приведения логической структуры базы данных к требуемому уровню нормальной формы. Существует шесть нормальных форм (НФ): первая НФ (1НФ), вторая НФ (2НФ), третья НФ (3НФ), нормальная форма Бойса-Кодда (БКНФ), четвертая НФ (4НФ), пятая НФ (5НФ). Каждая следующая нормальная форма должна удовлетворять требованиям предыдущей формы и некоторым дополнительным условиям. Ограничимся рассмотрением первых трех НФ, поскольку при практическом проектировании баз данных остальные НФ используются в редких случаях.

Первая нормальная форма

Таблица находится в 1НФ, если она удовлетворяет следующим требованиям:

а) не содержит полей с несколькими значениями;

б) ключевое поле не имеет пустот.

Первым требованием обеспечивается атомарность полей, т.е. каждое поле таблицы должно иметь только одно значение в каждой строке данных. Иногда неатомарное поле может быть представлено несколькими полями в виде повторяющейся группы, которая также запрещена в 1НФ. В процессе нормализации, необходимо повторяющиеся значения в полях превратить в повторяющиеся строки в других таблицах, а повторяющиеся группы в отдельные таблицы.

Вторым требованием через определение первичного ключа обеспечивается уникальность записей таблицы. Как правило, в 1НФ первичный ключ является составным.

Рассмотрим процесс приведения к 1НФ на примере. На рис. 1 представлен список таблиц проектируемой базы данных.

По определению логическая модель является отображением концептуальной
модели, в частности диаграммы классов. В данном случае исходными данными будут являться пять таблиц (Рис.1) , выделенных при построении концептуальной модели. Проанализируем их в соответствии с требованиями.

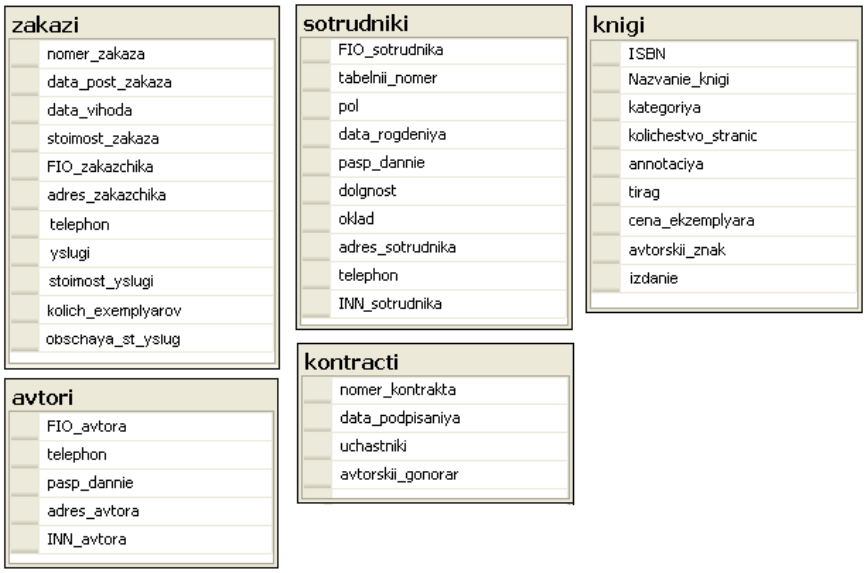

Рис.1– Структура таблиц базы данных «Издательство»

В таблицах на рис.1 не все поля удовлетворяют требованиям 1НФ. В частности, поле «adres_zakazchika» в таблице “Zakazi” не является атомарным, т.к. содержит информацию о городе, области, районе, улице , доме, квартире. В издательство обращаются заказчики из разных городов, впоследствии необходимо будет проводить анализ заказов по городам.

Поэтому целесообразно выделить информацию о городе в отдельное поле «gorod_zakazchika».А поле «adres_zakazchika» будет указывать улицу, дом и квартиру заказчика.

Поле «telephon» в таблице “Zakazi” содержит информацию о домашнем и контактном телефоне заказчика. Поэтому делим его на «dom_tel_zak»,«kont_tel_zak».

Поле «yslugi» в таблице “Zakazi” включает в себя услуги по оформлению книги(цветность страниц, качество обложки, формат книги) . Поле «stoimost_yslugi» в той же таблице включает перечень стоимости услуг. Т.е. эти поля образуют повторяющиеся группы. Выделим их в отдельную таблицу “Yslugi”.

Поле «adres_sotrudnika» в таблице “Sotrudniki” считаем атомарным, т.к. постоянная работа в издательстве подразумевает проживание сотрудника в городе, где находится издательство.

Поле «pasportnie_dannie» в таблице “Sotrudniki” содержит несколько значений – данные о серии, номере и дате выдачи документа, а также о наименовании государственного органа, выдавшего документ.

С целью поддержания целостности данных о сотрудниках целесообразно разбить данное поле на три поля: «sn_pasp_sotr», «data_pasp_sotr» и «mest _pasp_sotr».

Поле «telephon» в таблице “Sotrudniki” содержит информацию о домашнем и контактном телефоне сотрудника. Поэтому делим его на «dom_tel_sotrudnika»,«kont_tel_sotrudnika».

Поле «adres_avtora» в таблице “Avtori” разделим на два поля: «gorod_avtora», «adres_avtora»,т.к. с издательством могут сотрудничать авторы из разных городов.

Поле «pasportnie_dannie» в таблице “Avtori” является многозначным. Аналогично полю «pasportnie_dannie» в таблице “Sotrudniki” разбиваем это поле на три «sn_pasp_avt», «data_pasp_avt» и «mest_pasp_avt».

Также поле «telephon» в таблице “Avtori”, делим на «dom_tel_avtora»,«kont_tel_avtora».

В поле «uchastniki_kontrakta» в таблице “Kontrakti” выделим два поля «avtor_kontrakt» и «menedger_kontrakt», значения которых определяют участников заключения контракта. Но в ранее рассмотренном анализе предметной области отмечено, что контракт подписывается всеми авторами книги, а также, что для каждого автора устанавливается свой гонорар, в зависимости от порядкового номера, под которым указан автор. Таким образом поля «avtor_kontrakt» и «gonorar_avtora» являются многозначными. Поэтому необходимо ввести отдельную таблицу “Avtor_knigi”, в которой будут содержаться поля: «avtor_kontrakt», «gonorar_avtora», «poryadkovii_nomer» и поле «ISBN», указывающее на книгу за которую выплачивается гонорар.

Таким образом, получили список атомарных полей без повторяющихся групп, т.е. первое требование 1НФ выполнено.

Чтобы обеспечить второе требование необходимо определить первичные ключи в каждой таблице.

Первичный ключ в таблице “Knigi” -поле «ISBN», которое однозначно определяет остальные не ключевые поля данной таблицы.

В таблице “Sotrudniki” для первичного ключа выделим поле «tabelnii_nomer», приняв при этом допущение - каждому сотруднику присваивается свой табельный номер, а записи об уволенных сотрудниках деактивируются.

В таблице “Zakazi” выделяем составной первичный ключ из полей «Nomer_zakaza» и «FIO_zakazchika», однозначно определяющие остальные не ключевые поля данной таблицы.

В таблице “Kontrakti” в качестве первичного ключа выделим поле «nomer_kontrakta».

В таблицах “Yslugi”, «Avtori» и “Avtor_knigi” нет подходящих полей для первичного ключа, поэтому вводим в данные таблицы семантически незначащие поля «Id_yslugi», «Id_avtora» и «Id_avt_knigi» соответственно, которые однозначно определяют остальные не ключевые поля.

Поскольку ключевые поля всех таблиц не имеют пустот (Рис.2), то процесс приведения таблиц к 1НФ считаем завершенным.

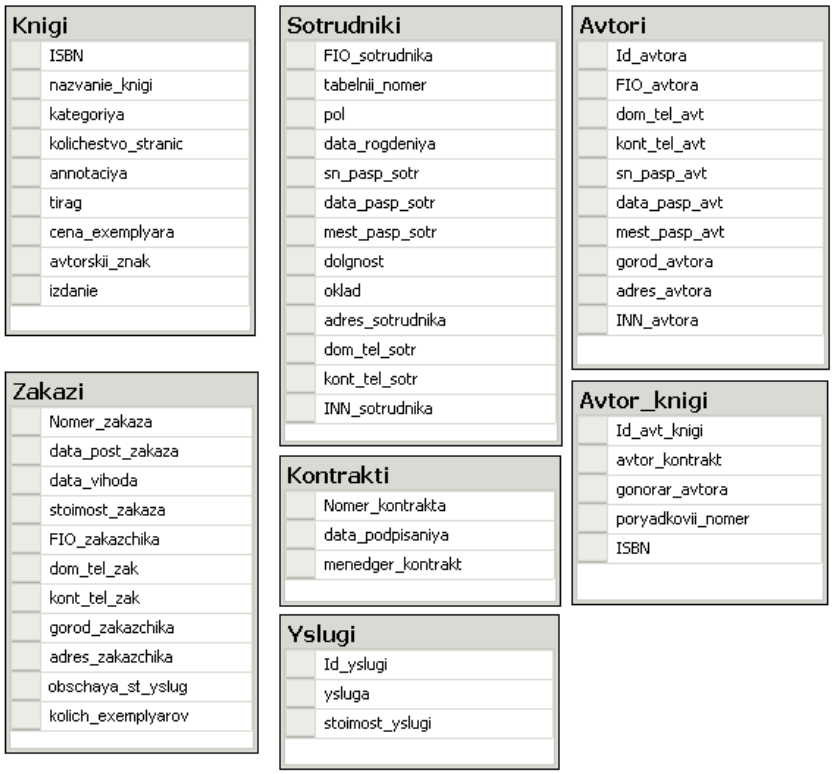

Рис. 2 – Структура таблиц в первой нормальной форме

Вторая нормальная форма

Таблица находится во 2НФ, если она удовлетворяет следующим требованиям:

а) таблица должна быть приведена к 1НФ;

б) поля, которые зависят только от части первичного ключа должны быть выделены в состав отдельных таблиц ;

в) все таблицы должны быть связаны между собой.

В рассматриваемом примере таблицы приведены к 1НФ, то есть одно из требований 2НФ выполнено. Для обеспечения второго требования, необходимо проанализировать каждую из полученных таблиц (Рис.2), на предмет зависимостей неключевых полей таблицы от части первичного ключа.

Так ключевые поля всех таблиц,представленных на рис.2,за исключением “Sotrudniki” и “Zakazi” , являются простыми и однозначно определяют детальные неключевые поля соответствующих таблиц. Эти таблицы оставляем без изменений.

В таблице “Sotrudniki” поле «oklad» зависит от не ключевого поля «dolgnost», поэтому необходимо создать отдельную таблицу “Dolgnosti” и перенести в нее поля «dolgnost» и «oklad». В качестве первичного ключа этой таблицы введем семантически незначащее поле «Id_dolgnosti».

В таблице “Zakazi” первичный ключ состоит из полей «Nomer_zakaza» и «FIO_zakazchika». Поэтому разделим эту таблицу на две “Zakazi” и “Zakazchiki”. В таблице “Zakazi” будут содержаться поля : «nomer_zakaza», «data_post_zakaza» , «data_vidachi», «stoimost_zakaza», которые характеризуют сделанный заказ. «Nomer_zakaza» будет первичным ключом.

Таблица “Zakazchiki” будет содержать поля : «FIO_zakazchika», «dom_tel_zak»,«kont_tel_zak»,«gorod_zakazchika»,«adres_zakazchika», характеризующие заказчика. Для обеспечения уникальности записей таблицы вводим семантически незначащее поле «Id_zakazchika» .

Второе требование 2НФ выполнено, перейдем к выполнению третьего требования.

В обязанности сотрудников входит редактирование книг, составление контрактов и принятие заказов. Таким образом, таблицы “Kontrakti”, “Knigi” и “Zakazi” должны быть связаны с таблицей “Sotrudniki”, т.е. внешним ключом в данных таблицах будет поле «tabelnii_nomer».

ISBN книги однозначно определяет редактируемую книгу, редакторов, авторов книги, подписанный контракт, менеджера, а также заказ. Поэтому в таблицах “Kontrakti”, “Avtor_knigi” и “Zakazi” добавим поле «ISBN»-внешний ключ,который будет являться полем связи с таблицей книги.

Таким образом получается,что в ряде таблиц присутствуют одинаковые поля связи,в результате чего появляется избыточность, а это противоречит основной цели нормализации –устранение избыточности данных.

Для решения этой задачисоздадим отдельную таблицу, называемую ассоциативной, “Rabota_sotr” . Данная таблица будет содержать общие поля «tabelnii_nomer», «ISBN», а также «vid_rabot»,для указания вида работ сотрудников. Первичным ключом в этой таблице назначим семантически не значащее поле «Id_rab_sotr».
Между тем каждую книгу могут редактировать несколько редакторов, т.е. одну и ту же работу могут выполнять несколько сотрудников, а один менеджер может принимать заказ и составлять контракт,т.е. один сотрудник может выполнять несколько видов работ. Поэтому необходимо вынести поле «vid_rabot» в отдельную таблицу. Введем семантически не значащее поле «Id_vid_rabot»-первичный ключ.
Каждому заказу соответствуют различные наборы услуг, поэтому образуется связь типа «многие-ко-многим». Эту проблему также можно решить с помощью ассоциативной таблицы. Добавим таблицу “Yslugi_knigi”, содержащую поля «Id_yslugi», «nomer_zakaza». В качестве первичного ключа введем семантически незначащее поле «Nomer_zapisi».

Таблицы “Knigi” и “Avtori” связаны между собой через таблицу “Avtor_knigi”. Поле «avtor_kontrakt» в таблице “Avtor_knigi” содержит информацию об уникальном номере автора, т.е. «Id_avtora». Поэтому заменим название поля «avtor_kontrakt» на «Id_avtora» и установим связь “Avtori”-“Avtor_knigi”, посредством поля «Id_avtora». При этом тип связи будет «один-ко-многим» , т.к. один автор может написать несколько книг.Установим связь “Knigi”-“Avtor_knigi” посредством поля «ISBN». Тип связи «один-ко-многим»,т.к. у одной книги может быть несколько авторов.

Далее переходим к таблице “Rabota_sotr”, посредством внешних ключей «ISBN» и «tabelnii_nomer» связываем ее с таблицами “Knigi” и “Sotrudniki” соответственно. Тип связей «один-ко-многим».
Также таблицу “Rabota_sotr” необходимо связать с таблицей “Zakazi” .Для этого добавляем в таблицу “Zakazi” внешний ключ «Id_rab_sotr» и устанавливаем связь один-ко-многим посредством этого поля.
Поле «menedger_kontrakt» в таблице “Kontrakti” содержит информацию содержит информацию о работе сотрудника,в данном случае менеджера-«Id_rab_sotr». Поэтому заменим название поля «menedger_kontrakt» на «Id_rab_sotr» и свяжем с таблицей “Rabota_sotr” .
Для установления связи с ранее выделенной таблицей “Vid_rabot” добавим в таблицу “Rabota_sotr” поле «Id_vid_rabot» - внешний ключ и проведем связь «один-ко-многим».

Таблицу “Sotrudniki” необходимо связать с таблицой “Dolgnosti”, поэтому добавляем в таблицу “Sotrudniki” поле «Id_dolgnosti»-внешний ключ и проведем связь типа «один-ко-многим» посредством этого поля.

Таблицы “Zakazi” и “Zakazchiki” должны быть связаны между собой, поэтому вводим в таблицу “Zakazi” поле «Id_zakazchika» и посредством этого поля проводим связь типа «один-ко-многим», т.к. один заказчик может сделать несколько заказов.
Далее связываем между собой таблицы “Zakaz” и “Yslugi_knigi”, для этого добавим в таблицу “Yslugi_knigi” внешний ключ «nomer_zakaza».
И наконец ,посредством поля «Id_yslugi» проводим связь “Yslugi”-“Yslugi_knigi”. Тип данных связей «один-ко-многим».
Поскольку все таблицы связаны между собой (Рис.3), то процесс приведения таблиц ко 2НФ считаем завершенным.

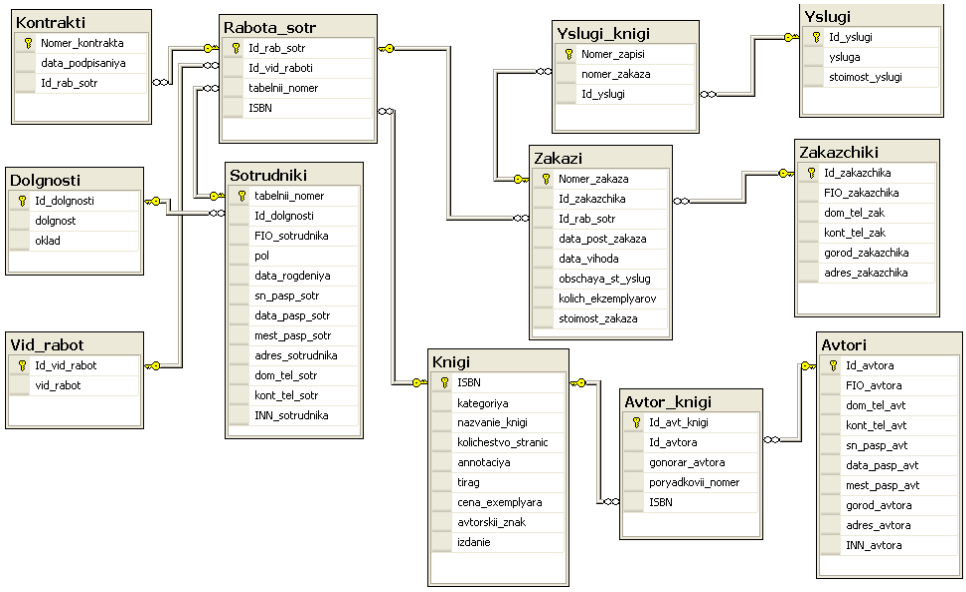

Рис.3- Структура таблиц во второй нормальной форме

Третья нормальная форма

Таблица находится в 3НФ, если она удовлетворяет следующим требованиям:

а) таблицы должны быть приведены ко 2НФ;

б) не должно быть транзитивных зависимостей между не ключевыми полями;

В данном примере таблицы приведены к первой и второй нормальным формам, то есть одно из требований выполнено. Перейдем к выполнению второго требования.

Транзитивная зависимость имеет место, если какое-либо неключевое поле функционально зависит от другого неключевого поля, а тот в свою очередь функционально зависит от ключа.

В нашем случае поле «kategoriya» в таблице “Knigi” определяется только ISBN книги, поэтому одна и таже категория будет многократно повторяться в разных экземплярах данного информационного объекта. При этом наблюдаются затруднения в корректировке , например, при удалении какой-

либо категории или добавлении новой,а также неоправданное использование памяти для хранения дублированной информации.
Для устранения транзитивной зависимости необходимо провести "расщепление" исходной таблицы. В результате расщепления часть реквизитов удаляется из исходной таблицы и включается в состав других (возможно, вновь созданных) таблиц.
Добавим таблицу “Kategorii” ,которая будет содержать поля «name_kategorii» и «Id_kategorii» - первичный ключ.
Далее добавим внешний ключ «Id_kategorii» в таблицу “Knigi” и проведем связь “Kategorii”-“Knigi”,тип связи «один-ко-многим».

Поле «stoimost_zakaza» в таблице “Zakaz” однозначно определяется полями «obschaya_st_yslug» , «kolich_ekzemplyrov» и «cena_ekzemplyara», то исключаем его из таблицы.
Поле «obschaya_st_yslug» расчитывается исходя из перчня «yslug» с указанием стоимости каждой ,определенного в конкретном заказе. Поэтому исключаем его из таблицы.
Поскольку в таблицах не осталось транзитивных зависимостей (Рис.4), то процесс приведения таблиц к ЗНФ считаем завершенным.

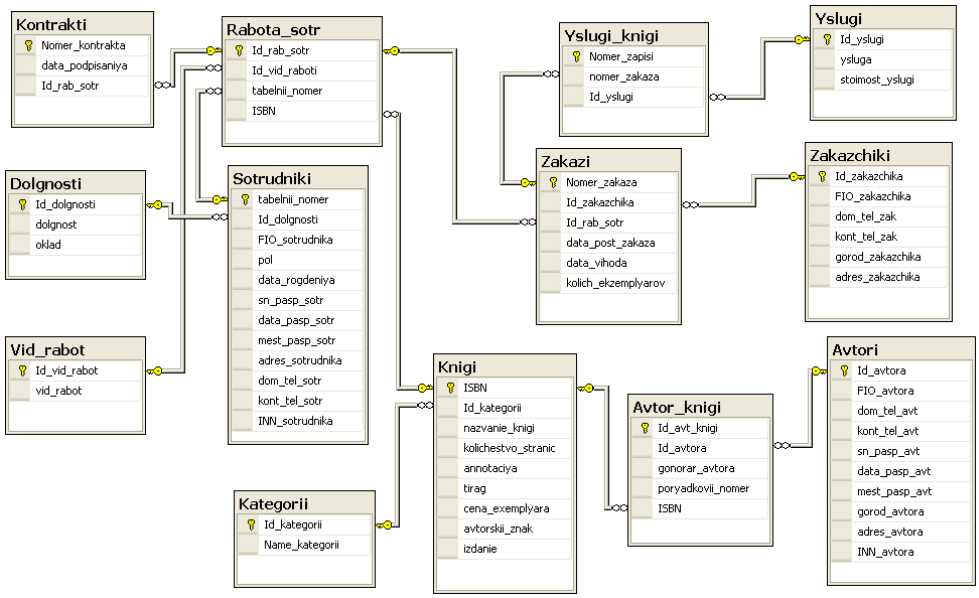

Рис.4- Структура таблиц в третьей нормальной форме

Таким образом, мы получаем корректную, нормализированную структуру базы данных.