Семинар 1 – **Системы управления базами данных в …**



 Семинар 2 - **Концептуальное моделирование предметной области**

Рекомендации к Упражнению 1 - 2:

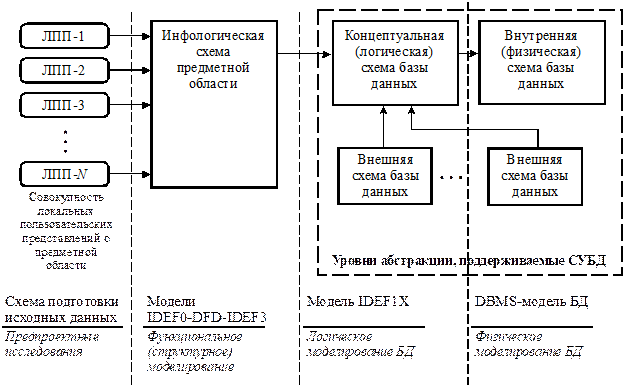
* 1. Определить, какие предметы реального мира необходимо абстрагировать в виде объектов учета.
  2. Сформировать перечень значимых свойств объектов учета.
  3. Подобрать краткие, но емкие имена для объектов учета и их значимых свойств.

Подсказка к Упражнению 1 - 3: Модели данных: иерархическая, сетевая, реляционная, объектно-реляционная, объектная.

Бизнес-правило - это описание ограничения на хранимые данные, обусловленное особенностями предметной области.

Рекомендация к Упражнению 2 - 1: Имя существительное абстрагирует по тексту предмет, процесс, явление реального мира или его свойство. Оно указывает на возможное отношение (сущность) или атрибут отношения. Найти имя существительное и определить, что оно абстрагирует: объект учета или значимое свойство объекта учета. Если это абстракция объекта учета, значит Вы имеете дело с отношением (будущей таблицей базы данных), а если рассматривается значимое свойство объекта учета - то это атрибут (будущая колонка таблицы).

Концептуальная схема процесса создания базы данных



ЛПП - локальное пользовательское представление о предметной области

Инфологическая схема (ИЛМ) предметной области - обобщение ЛПП. Варианты представления:

* описание в виде текста;
* диаграмма в нотации ERD, IM или иной, доступной рабочей группе;
* список заголовков сущностей (отношений) на псевдокоде.

**Концептуальная** (логическая) схема (модель) базы данных - формализованное описание совокупности объектов учета в терминах выбранной модели данных без учета особенностей СУБД, под управлением которой база данных будет реализована физически.

**Внутренняя** (физическая) схема (модель) базы данных - это логическая схема базы данных, учитывающая требования, особенности и ограничения СУБД, выбранной для физической реализации.

Семинар 3 – **Основные положения реляционной модели данных**



Объект учета отображается в концептуальной модели в виде прямоугольника:

Имя объекта учета записывается внутри прямоугольника существительным в единственном числе именительном падеже: **Студент, Департамент, Автомобиль, Обучение, Изготовление радиатора, Пошив платья,...**

Покупа тель 
Зекез 
Товар 
Покупке 

ОБЪЕКТ УЧЕТА -> СУЩНОСТЬ -> ОТНОШЕНИЕ -> ТАБЛИЦА

СВОЙСТВО -> АТРИБУТ -> АТРИБУТ -> КОЛОНКА



Подсказки

Виды ключей реляционного отношения: потенциальный

суперключ

первичный

альтернативный (уникальный)

внешний

инверсионный вход

ключ отношения





Семинар 4 – **Связи отношений в реляционной модели данных**



**ПОДСКАЗКИ**

Бизнес-сценарий -- это агрегированное (обобщенное) из локальных пользовательских представлений описание предметной области. Описание представляется в виде текста от третьего лица. В описании должны содержаться указания на будущие объекты учета, их значимые свойства (характеристики) и предполагаемые ключевые фразы связи. Из описания можно получить указания для определения мощности (кардинальности) связи.

**Домашнее задание:**

1. В экспедиции телеграфа учитываются исходящие и входящие телеграммы. Каждая телеграмма имеет получателя и отправителя – абонентов телеграфа, характеризуется датой отправки и получения. Телеграфист обслуживает абонентов по отправке и приему телеграмм. Любой абонент телеграфа может быть обслужен любым телеграфистом. Функции телеграфиста может выполнять сотрудник, занимающий любую должность из утвержденного перечня телеграфа. Для доставки телеграмм в телеграфе учитываются места размещения абонентов (здание, кабинет, телефон).

Разработать информационную модель.

1. Д а н о: Множество *А* – террористических организаций (названия), например *А=*{*Эскада, Аль-Каида, Армия Джихада, Бригады мучеников Аль-Аксы*}, и множество *В* – известных экстремистов, например *В=*{*Бен-Ладен, Ш. Басаев, А. Джазир*}. Имеется множество свершившихся террористических актов, характеризуемых датой, местом, описанием, числом жертв. Место проведения террористического акта, в свою очередь, характеризуется страной и географическим пунктом. Свойства описания террористического акта – это вид (взрыв, захват заложников, катастрофа и т. п.), наименование дополнительной характеристики, значение дополнительной характеристики. Перечень дополнительных характеристик зависит от вида террористического акта.

Разработать информационную модель предметной области.

Семинар 5 – **Разрешение связей типа «многие-ко-многим»**





ПОДСКАЗКИ:

1. Связь "супертип-подтип" преобразуется через обобщения (обобщающие свойства) экземпляров отношения. Эквивалентная схема должна быть регулярной структурой. Нерегулярные структуры приводят к потере целостности данных!
2. Дуговая связь приводит к нарушению целостности данных. Она должна быть преобразована в эквивалентную путем отыскания обобщающего свойства, как и в случае с связью "супертип-подтип".

Семинар 6 – **Нормализация отношений**



Дано:

Авиабила 
* пассмира 
• паспорт пассажира 
* N2 рейса 
о тариф 
* стоимость 
о направление 
о самолет 
* дата вылета 
* время вылета 
* Аэропорт вылета 
* дата прмета 
* время прмета 
* Аэропорт прилета 

Привести отношение к первой нормальной форме.

Bring the relation to first normal form.

® Отношение считается приведенным к первой нормальной форме (1НФ) тогда и только тогда, когда все его атрибуты атомарны и сформированы все возможные минимальные потенциальные ключи.

A relation is considered to be reduced to the first normal form (1NF) if and only if all its attributes are atomic and all possible minimal candidate keys are formed.

® Атомарный атрибут - это атрибут, который не возможно разделить на податрибуты, каждый из которых имеет смысл.

An atomic attribute is an attribute that cannot be divided into sub-attributes, each of which has a meaning.

® Потенциальный ключ - это атрибут или совокупность атрибутов, которые комбинацией своего значения однозначно отличают кортеж от других кортежей (идентифицируют экземпляр объекта учета).

A candidate key is an attribute or a set of attributes that uniquely distinguishes a tuple from other tuples by a combination of its value (identifies an instance of an accounting object).

Каждый потенциальный ключ должен быть минимальным. Это означает, что если из него исключить любой, входящий в его состав атрибут, то он утрачивает свойство идентификации кортежа.

Each candidate key must be minimal. This means that if any attribute included in it is excluded from it, then it loses the tuple identification property.

Решение:

Solution:

ПК1 
ПК2 
пка 
пка 
пкз 
пка 
Авиабилет 
* ф И пассажира 
• Паспорт. серия пассажир 
* Паспортномер пассажира 
* NQ рейса 
о Тариф 
* Стоимость 
о Направление 
о Самолетпроиваодитель 
о Самолетмодель 
* Дата и время вылета 
* Аэропорт вылета 
* Дата и время прилета 
* Аэропорт прилета 
Формально должно быть: 
• Фамилия 
* имя 
* отчество 
Атрибут, содержащий дату и время, 
атомарен по сваеи природе и 
не может быть разделен на податриву'ты - 
хранит целое число секунд от условного 
начала отсчета времени! 

Семинар 7 – **Вторая и третья нормальные формы отношения**



Second normal form (2NF).

® A relation is considered reduced to second normal form (2NF) if and only if it has already been reduced to first normal form (1NF) and all non-key (non-key) attributes are functionally fully dependent on all potential keys.

® A non-key attribute is an attribute that is not part of any candidate key.

® An attribute is functionally fully dependent on the candidate key when it depends on it at all, and does not depend on any of its constituent parts (any subset of the candidate key's attributes). Full functional dependence predetermines that when any attribute is excluded from the candidate key, the non-key attribute no longer depends on the candidate key transformed in this way.

® Reduction to a complete functional dependence requires that a non-key attribute generally functionally depend on the potential key, and then also completely (on the entire set of attributes of the potential key). ® If you change the value of the determinant (identifier of the accounting object instance), then the dependent part may or may not change. When changing the value of the dependent part, in the presence of a functional dependence, the value of the determinant must also change. If there is a situation in which the value of the determinant does not change, then this means that the desired functional dependence does not exist.

Every non-key attribute that is not functionally dependent on a candidate key must be excluded from the relation. Each non-key attribute that is functionally dependent on a candidate key part must be excluded from the normalizable relation along with that part of the composite candidate key. All attributes excluded from the relation are placed in new relations to be normalized.

Вторая нормальная форма (2НФ).

® Отношение считается приведенным ко второй нормальной форме (2НФ) тогда и только тогда, когда оно уже приведено к первой (1НФ) и все не ключевые (non-key) атрибуты функционально полно зависят от всех потенциальных ключей.

® Не ключевой атрибут - это атрибут, который не входит в состав никакого потенциального ключа.

® Атрибут функционально полно зависит от потенциального ключа тогда, когда он, вообще от него зависит, и не зависит ни от какой его составной части (любого подмножества атрибутов потенциального ключа). Полная функциональная зависимость предопределяет, что при исключении из потенциального ключа любого атрибута, не ключевой атрибут более не зависит от преобразованного таким образом потенциального ключа.

® Приведение к полной функциональной зависимости требует, чтобы не ключевой атрибут вообще функционально зависел от потенциального ключа, а затем еще и полно (от всей совокупности атрибутов потенциального ключа).

® Если изменить значение детерминанта (идентификатора экземпляра объекта учета), то зависимая часть может измениться или нет. При изменении значения зависимой части, при наличии функциональной зависимости, значение детерминанта должно также измениться. Если существует ситуация, при которой значение детерминанта не изменяется, то это означает, что искомая функциональная зависимость не существует.

Каждый не ключевой атрибут, не состоящий в функциональной зависимости от потенциального ключа, должен быть исключен из отношения. Каждый не ключевой атрибут, состоящий в функциональной зависимости от части потенциального ключа, должен вместе с этой частью составного потенциального ключа быть исключен из нормализуемого отношения. Все исключаемые из отношения атрибуты помещаются в новые отношения, подлежащие нормализации.

**Life hack (engineering technique):**

If you change the value of the determinant (potential key), then the value of the dependent part (not the key attribute) may change or remain the same. If you change the value of the dependent part and there is a functional dependence, then the value of the determinant cannot but change. Otherwise, there is no functional dependency!

**Лайфхак (инженерный прием):**

Если изменить значение детерминанта (потенциального ключа), то значение зависимой части (не ключевого атрибута) может измениться, а может остаться прежним. Если изменить значение зависимой части и имеется функциональная зависимость, то значение детерминанта не может не измениться. Иначе функциональная зависимость отсутствует!

ПК1 
ПК2 
пка 
пка 
пкз 
пка 
Авиабилет 
* ф пассажира 
• Паспорт. серия пассажира 
* Паспортномер пассажира 
* NQ рейса 
о Тариф 
* Стоимость 
о Направление 
о Самолетпроиваодитель 
о Самолетмодель 
* Дата и время вылета 
* Аэропорт вылета 
* Дата и время прилета 
* Аэропорт прилета 

Если на рейсе заменили самолет, то это никак не влияет на предстоящий полет и значимые свойства авиабилета!

If the plane was replaced on the flight, then this does not affect the upcoming flight and the significant properties of the ticket!

Атрибуты, не состоящие в функциональной зависимости от потенциальных ключей, должны быть перенесены в новое отношение, которое должно быть нормализовано позже.

Атрибуты, состоящие в не полной (частичной) функциональной зависимости, должны быть вместе с частью потенциального ключа перенесены в новое отношение, которое подлежит нормализации позже!

Attributes that are not functionally dependent on candidate keys must be transferred to a new relation, which must be normalized later. Attributes that are not fully (partially) functionally dependent must be transferred, together with a part of the potential key, to a new relationship, which must be normalized later!

Третья нормальная форма (3НФ).

® Отношение считается приведенным ко третьей нормальной форме (3НФ) тогда и только тогда, когда оно уже приведено ко второй (2НФ) и все не ключевые (non-key) атрибуты функционально нетранзитивно зависят от всех потенциальных ключей.

® Не ключевой атрибут - это атрибут, который не входит в состав никакого потенциального ключа.

® Если не ключевой атрибут функционально зависит от потенциального ключа, а другой не ключевой атрибут функционально зависит от первого не ключевого атрибута, то это означает, что второй не ключевой атрибут функционально зависит от того же потенциального ключа. Атрибут функционально нетранзитивно зависит от потенциального ключа тогда, когда он, вообще от него зависит, и не зависит ни от какого другого не ключевого атрибута.

® Если изменить значение детерминанта (одного из не ключевых атрибутов), то зависимая часть (другой не ключевой атрибут) может измениться или нет. При изменении значения зависимой части, при наличии функциональной зависимости, значение детерминанта должно также измениться. Если существует ситуация, при которой значение детерминанта не изменяется, то это означает, что искомая функциональная зависимость не существует.

Каждая пара не ключевых атрибутов, состоящих в функциональной зависимости друг от друга (не обязательно взаимно), должна быть исключена из отношения. Все исключаемые из отношения атрибуты помещаются в новые отношения, подлежащие нормализации.

Third normal form (3NF).

® A relation is considered to be reduced to third normal form (3NF) if and only if it has already been reduced to second normal form (2NF) and all non-key (non-key) attributes are functionally non-transitively dependent on all candidate keys.

® A non-key attribute is an attribute that is not part of any candidate key.

® If a non-key attribute is functionally dependent on the candidate key, and another non-key attribute is functionally dependent on the first non-key attribute, then this means that the second non-key attribute is functionally dependent on the same candidate key. An attribute is functionally non-transitively dependent on a candidate key when it is dependent on it at all, and does not depend on any other non-key attribute.

® If you change the value of the determinant (one of the non-key attributes), then the dependent part (other non-key attribute) may or may not change. When changing the value of the dependent part, in the presence of a functional dependence, the value of the determinant must also change. If there is a situation in which the value of the determinant does not change, then this means that the desired functional dependence does not exist.

Each pair of non-key attributes that are functionally dependent on each other (not necessarily mutually) must be excluded from the relationship. All attributes excluded from the relation are placed in new relations to be normalized.

**Life hack (engineering technique):**

Form all possible pairs of non-key attributes. Identify the functional relationship between the attributes of each of the pairs, in turn considering one of them, then the other, as a determinant. Remove all pairs in which a functional dependence is found from the relationship.

**Лайфхак (инженерный прием):**

Сформировать все возможные пары из не ключевых атрибутов. Выявить функциональную зависимость между атрибутами каждой из пар, по очереди считая детерминантом то один из них, то другой. Все пары, в которых обнаружена функциональная зависимость удалить из отношения.

ПК1 
ПК2 
пка 
пка 
пкз 
пка 
Авиабилет 
* ф И пассажира 
• Паспорт. серия пассажира 
* Паспортномер пассажира 
* NQ рейса 
о Тариф 
* Стоимость 
о Направление 
* Дата и время вылета 
* Аэропорт вылета 
* Дата и время прилета 
* Аэропорт прилета 
стоимость определяется, как сумма 
тарифа и надбавок за обслуживание 
в аэропортах, топливный сбор и 
страховкьъ следовательно, стоимость 
функционально зависит от тарифа 

Семинар 8 – **НФБК и четвертая нормальная форма**



**Нормальная форма Бойса - Кодда (усиленная 3НФ)**

® Отношение считается приведенным к нормальной форме Бойса - Кодда (НФБК) тогда и только тогда, когда оно уже приведено к 3НФ и каждый его детерминант является только потенциальным ключом.

Если атрибут составного потенциального ключа функционально зависит от не ключевого атрибута, то такая зависимость называется инверсной (обратной) и она должна быть исключена из отношения, приведенного в НФБК.

Boyce-Codd normal form (strengthened 3NF)

® A relation is considered to be reduced to Boyce-Codd Normal Form (BCNF) if and only if it has already been reduced to 3NF and each of its determinants is only a potential key.

If an attribute of a composite candidate key is functionally dependent on a non-key attribute, then such dependence is called inverse (reverse) and it should be excluded from the relationship given in the BCNF.

**Life hack (engineering technique):**

To achieve BCNF, it is necessary to compose all possible pairs of attributes, where not the key attribute is considered as the determinant, but the attribute of the composite candidate key as the dependent part. Check for functional dependence in all pairs. Pairs in which a functional dependence will be found should be excluded from the relationship.

**Лайфхак (инженерный прием):**

Для достижения НФБК необходимо составить все возможные пары атрибутов, где в качестве детерминанта рассматривается не ключевой атрибут, а в качестве зависимой части атрибут составного потенциального ключа. Проверить наличие функциональной зависимости во всех парах. Пары, в которых будет обнаружена функциональная зависимость исключить из отношения.

ПК1 
ПК2 
пка 
пка 
пкз 
пка 
Авиабилет 
* ф пассажира 
• Паспорт. серия пассажира 
* Паспортномер пассажира 
* NQ рейса 
о Направление 
* Дата и время вылета 
* Аэропорт вылета 
* Дата и время прилета 
* Аэропорт прилета 
интуишвно понятно, что дата и время 
выпета функционально зависят от 
номера рейса имеется 
функциональная зависимость 
номер рейса дата и время вылета 

The ARRIVAL DATE AND TIME attribute is also functionally dependent on the non-key attribute FLIGHT NUMBER. Therefore, it must be excluded from the relationship.

Атрибут ДАТА И ВРЕМЯ ПРИЛЕТА также состоит в функциональной зависимости от не ключевого атрибута НОМЕР РЕЙСА. Следовательно, его нужно исключить из отношения.

Fourth normal form (4NF)

® A relation is considered to be reduced to fourth normal form if and only if it has already been reduced to BCNF and does not contain non-trivial multivalued dependencies.

**Четвертая нормальная форма (4НФ)**

® Отношение считается приведенным к четвертой нормальной форме тогда и только тогда, когда оно уже приведено к НФБК и не содержит нетривиальных многозначных зависимостей.

With respect to the AIR TICKET, after the exclusion of attributes that do not meet the requirements of the BCNF, a situation has arisen in which it is necessary to re-generate candidate keys. Otherwise, each passenger will be counted only in conjunction with a single flight direction.

В отношении АВИАБИЛЕТ после исключения атрибутов, не соответствующих требованиям НФБК, образовалась ситуация, в которой необходимо вновь сформировать потенциальные ключи. Иначе каждый пассажир будет учтен только в совокупности с единственным направлением перелета.

пк1 
Авиабилет 
* ф пассажира 
• Паспорт серия пассажира 
* Паспортномер пассажира 
о Направление 
* Аэропорт вылета 
* Аэропорт прилета 

Многократное повторение кортежей с одинаковым значением Ф.И.О. пассажира (а также номером и серией паспорта) является следствием многозначной зависимости атрибутов НАПРАВЛЕНИЕ, АЭРОПОРТ ВЫЛЕТА и АЭРОПРОРТ ПРИЛЕТА от атрибутов Ф.И.О. и ПАСПОРТ.СЕРИЯ ПАССАЖИРА совместно с ПАСПОРТ.НОМЕР ПАССАЖИРА.

Multiple repetition of tuples with the same value passenger (as well as the number and series of the passport) is a consequence of the multi-valued dependence of the attributes DIRECTION, AIRPORT OF DEPARTURE and AIRPORT OF ARRIVAL on the attributes of F.I.O. and PASSPORT.SERIES OF THE PASSENGER together with PASSPORT.NUMBER OF THE PASSENGER.

**Life hack (engineering technique):**

To achieve 4NF, it is necessary, according to the rules of the "entity-relationship" methodology, to clarify how many accounting objects are simultaneously abstracted in a normalizable relation. If more than one accounting object is found, then it is necessary to decompose the relation into sub-relations in accordance with the found accounting objects.

**Лайфхак (инженерный прием):**

Для достижения 4НФ необходимо по правилам методологии "сущность-связь" уточнить сколько объектов учета одновременно абстрагировано в нормализуемом отношении. Если найдено более одного объекта учета, то необходимо отношение декомпозировать на подотношения в соответствие с найденными объектами учета.

Пассажир 
* ф пассажира 
Паспорт. серия пассажира 
* Паспортномер пассажира 
Направление перелета 
о Населенный пункт 
* Аэропорт вылета 
* Аэропорт прилета 

Семинар 9 – **Пятая нормальная форма и ДКНФ**



**Пятая нормальная форма (5НФ, проекционно-соединительная нормальная форма - ПСНФ)**

® Отношение считается приведенным к пятой нормальной форме (5НФ) тогда и только тогда, когда оно уже приведено к 4НФ и не содержит зависимости соединения.

Зависимость соединения порождает эффект декомпозиции с соединением (JOIN) с потерей целостности результата (возникновение ложных кортежей).

Fifth normal form (5NF, projective-connective normal form - PSNF)

® A relation is considered to be reduced to fifth normal form (5NF) if and only if it has already been reduced to 4NF and does not contain a join dependency.

Join dependency generates the effect of decomposition with a join (JOIN) with loss of integrity of the result (the appearance of false tuples).

**Направление перелета**

|  |  |  |
| --- | --- | --- |
| **Направление** | **Аэропорт вылета** | **Аэропорт прилета** |
| Москва | ULV | DME |
| Москва | ULY | SVO |

ULV – Аэропорт г. Ульяновска "Баратаевка"

ULY – Аэропорт г. Ульяновска "Восточный"

После декомпозиции (разделения на две таблицы)

|  |  |
| --- | --- |
| **Направление** | **Аэропорт вылета** |
| Москва | ULV |
| Москва | ULY |

И

|  |  |
| --- | --- |
| **Направление** | **Аэропорт прилета** |
| Москва | DME |
| Москва | SVO |

Для представления данных по запросу пользователя выполняется соединение (операция JOIN):

|  |  |  |
| --- | --- | --- |
| **Направление** | **Аэропорт вылета** | **Аэропорт прилета** |
| Москва | ULV | DME |
| Москва | ULV | SVO |
| Москва | ULY | DME |
| Москва | ULY | SVO |

Красным цветом показаны ложные кортежи, нарушающие целостность, появившиеся в следствие наличия в новых, полученных после декомпозиции отношения НАПРАВЛЕНИЕ ПЕРЕЛЕТА, отношениях зависимости соединения (НАПРАВЛЕНИЕ – АЭРОПОРТ ВЫЛЕТА и НАПРАВЛЕНИЕ – АЭРОПОРТ ПРИЛЕТА).

After decomposition (split into two tables)

To represent data at the user's request, a join is performed (JOIN operation):

The red color shows false tuples that violate the integrity, which appeared as a result of the presence in the new, obtained after the decomposition of the FLIGHT DIRECTION relation, connection dependency relations (DIRECTION - DEPARTURE AIRPORT and DIRECTION - ARRIVAL AIRPORT).

The solution of the decomposition problem must be carried out taking into account the exclusion of the connection dependency in the new relations. This is possible in the case when the properties of one accounting object are concentrated in a single relation, and not "smeared" over several.

Решение задачи декомпозиции должно выполняться с учетом исключения в новых отношениях зависимости соединения. Это возможно в том случае, когда свойства одного объекта учета сосредоточены в единственном отношении, а не "размазаны" по нескольким.

**Лайфхак (инженерный прием):**

Для достижения 5НФ необходимо проверить, чтобы все атрибуты, абстрагирующие значимые свойства объекта учета были сосредоточены в рамках единственного отношения, абстрагирующего сам этот объект учета. Что и требуется в методологии П.Чена "сущность-связь". Если будет обнаружено, что значимые свойства объекта учета абстрагируются атрибутами любых других отношений, то необходимо их перенести в "родное" отношение.

СОЕДИНЕНИЕ ДОЛЖНО ПРОИЗВОДИТЬСЯ НАД ОБЪЕКТАМИ УЧЕТА, А НЕ НА УРОВНЕ ОТДЕЛЬНЫХ ИХ СВОЙСТВ!

**Life hack (engineering technique):**

To achieve 5NF, it is necessary to check that all attributes that abstract the significant properties of the accounting object are concentrated within a single relationship that abstracts this accounting object itself. This is what is required in P. Chen's "entity-relationship" methodology.

If it is found that the significant properties of the accounting object are abstracted by the attributes of any other relations, then it is necessary to transfer them to the "native" relation. CONNECTION SHOULD BE CARRIED OUT OVER OBJECTS OF ACCOUNTING, AND NOT AT THE LEVEL OF THEIR INDIVIDUAL PROPERTIES!

**Доменно-ключевая нормальная форма (ДКНФ)**

® Отношение считается приведенным к ДКНФ тогда и только тогда, когда оно уже приведено к 5НФ (ПСНФ), каждый его атрибут определен на одном из стандартных доменов (СТРОКА, ЧИСЛО, ДАТАВРЕМЯ, BLOB) и назначен первичный ключ.

Уточнение типов данных на базе стандартных доменов производится на этапе формирования физической структуры базы данных, исходя из возможностей выбранной СУБД (Свойство ограничения целостности отношения, требующее, чтобы тип данных атрибута был сформирован путем наложения ограничений на стандартный домен средствами встроенного в СУБД языка манипулирования данными, например, SQL).

Первичный ключ должен быть компактным (минимальное число атрибутов, лучше если только один), защищенным от аномалий вставки (BLOB не допустим, STRING не целесообразен, кроме случаев регулярной строки с жесткими правилами ее формирования, например, табельный номер, серийный номер, MAC-адрес и тому подобное), малоемким (требующим как можно меньше памяти для хранения).

Хороший первичный ключ - ЧИСЛО (защищен от аномалий вставки, малоемкий, обычно 4 байта), ему немного уступает по емкости домен ДАТАВРЕМЯ. Строковый первичный ключ пригоден только в случае, если строка регулярная и применяется жесткий шаблон ее формирования, например, номер учебной группы, личный номер военнослужащего, МАС-адрес устройства вычислительной техники и тому подобное.

В случае, если в модели отсутствует пригодный потенциальный ключ, целесообразно сформировать суррогатный ключ, основанный на домене ЧИСЛО - идентификатор объекта учета. Популярность приобрел тип **UUID** (англ. *universally unique identifier* «универсальный уникальный идентификатор») — стандарт идентификации, используемый в создании программного обеспечения, стандартизированный Open Software Foundation (OSF) как часть DCE — среды распределённых вычислений.

Domain Key Normal Form (DKNF)

® A relation is considered to be reduced to DKNF if and only if it is already converted to 5NF (PSNF), each of its attributes is defined on one of the standard domains (STRING, NUMBER, DATETIME, BLOB) and a primary key is assigned.

Specification of data types based on standard domains is performed at the stage of forming the physical structure of the database, based on the capabilities of the selected DBMS (Relation integrity constraint property requiring that the attribute data type be formed by imposing restrictions on the standard domain using the data manipulation language built into the DBMS, for example , SQL).

The primary key should be compact (minimum number of attributes, better if only one), protected from insertion anomalies (BLOB is not allowed, STRING is not appropriate, except in cases of a regular string with strict rules for its formation, for example, personnel number, serial number, MAC address and the like), low-capacity (requiring as little memory as possible for storage).

A good primary key is NUMBER (protected from insertion anomalies, low capacity, usually 4 bytes), it is slightly inferior in capacity to the DATETIME domain. The string primary key is suitable only if the string is regular and a rigid pattern of its formation is used, for example, the number of the study group, the personal number of the serviceman, the MAC address of the computer equipment, and the like.

If there is no suitable potential key in the model, it is advisable to generate a surrogate key based on the NUMBER domain - accounting object identifier. The UUID (universally unique identifier) type has gained popularity - an identification standard used in software development, standardized by the Open Software Foundation (OSF) as part of the DCE - distributed computing environment.

**Лайфхак (инженерный прием):**

Для исключения сложностей нормализации, получения максимально качественной заготовки, зачастую не требующей преобразований на уровне НФБК, 4НФ и 5НФ, необходимо выполнить качественное информационное моделирование, где безошибочно определить все объекты учета, их значимые свойства, абстрагировать их отношениями и атрибутами соответственно. Связать отношения между собой, разрешить связи типа "многие-ко-многим". Полученный отношения подвергнуть нормализации. Сформированная конструкция представляет собой логическую структуру базы данных, которую можно начинать преобразовывать в физическую на платформе выбранной СУБД.

**Life hack (engineering technique):**

To eliminate the difficulties of normalization, obtaining the highest quality blank, which often does not require transformations at the level of BNF, 4NF and 5NF, it is necessary to perform high-quality information modeling, where all accounting objects, their significant properties can be accurately identified, abstracted by their relationships and attributes, respectively. Associate relationships with each other, allow many-to-many relationships. The resulting relationship is subjected to normalization. The generated design is a logical database structure that can be converted into a physical one on the platform of the selected DBMS.

Задачи для решения на занятии и дома:

Дано:

|  |  |  |
| --- | --- | --- |
| Студент | Преподаватель | Учебная группа |
| Самсонов В.А. | Говоронков П.А. | ПИ21-1 |
| Трапаренков В.П. | Новиков-Прибой С.И. | ПИ21-1 |

Декомпозировать на СТУДЕНТ - УЧЕБНАЯ ГРУППА и ПРЕПОДАВАТЕЛЬ - УЧЕБНАЯ ГРУППА

Выполнить соединение по общему атрибуту УЧЕБНАЯ ГРУППА, найти нарушение целостности.

Дано:

|  |  |  |
| --- | --- | --- |
| Сотрудник | Должность | Отдел |
| Чайковский П.И. | аналитик | трейдерский |
| Гоголь Н.В. | аналитик | теханализа |

Декомпозировать на СОТРУДНИК - ДОЛЖНОСТЬ и ОТДЕЛ - ДОЛЖНОСТЬ

Выполнить соединение по общему атрибуту ДОЛЖНОСТЬ, найти нарушение целостности.

Дано:

|  |
| --- |
| **Документ**    \* Наименование  \* Дата исполнения  о Дата утверждения  \* Инвентарный номер  \* Архивный номер |
|  |

Найти потенциальные ключи, создать (выбрать) первичный ключ, определить атрибуты на стандартных доменах.

Дано:

|  |
| --- |
| **Абонент**    \* Ф.И.О.  \* Телефон |
|  |

Найти потенциальные ключи, создать (выбрать) первичный ключ, определить атрибуты на стандартных доменах.

Семинар 10 – **Физическая структура базы данных (миграция ключа отношения, оптимизация базы данных)**

Apex на сервере Финуниверситета [https://apex.fa.ru](https://apex.fa.ru/)

Имя пользователя - адрес почты без доменной части @fa.ru  
Пароль - от почтовой учетной записи  
Имя рабочей области (workspace) - имя пользователя с приставкой WS\_ (пример - WS\_AVOSIPOV)

Go to the SQL Workshop tab and launch the SQL language interpreter - SQL Commands. Enter the SQL statement SELECT \* FROM user\_tables; Select it and press the RUN button. In the Results tab, view the list of user schema tables.

Перейти во вкладку SQL Workshop и запустить интерпретатор языка SQL – SQL Commands.

Ввести SQL-предложение SELECT \* FROM user\_tables; Выделить его и нажать кнопку RUN.

Во вкладке Results просмотреть список таблиц схемы пользователя.

Логическая модель 
Модель автомобиля 
# Идентификатор модели 
* Производитель 
* Модель 
о Страна 
Физическая модель 
V Model 
* Producer 
* Model 
о Country 
CREATE ТАВИ 
mode1 (1 D М SMALLINT NOT NULL, 
ALTER 
ТАВИ 
Producer VARCHAR2 (25) NOT NULL, 
Mode1 VARCHAR2 (30) NOT NULL, 
Country CHAR (2) ) ; 
mode1 ADD PRIMARY КЕМ (1 D М) ; 

|  |  |
| --- | --- |
| Логическая модель  Logic model | Физическая структура (модель)  Physical structure |
| Отношение (Relation) | Таблица (Table) |
| Атрибут (Attribute) | Столбец (Column) |
| Ключ (Key) | Индекс (Index) |
| Связь (Relationship) | Первичный ключ - Внешний ключ (Primary Key – Foreign Key) |

**Виды ключей реляционной модели данных:**

Потенциальный ключ (ПК);

Первичный ключ (РК);

Альтернативный ключ (АК): запрет повторяющихся значений атрибута, сортировка по возрастанию или по убыванию;

Инверсионный вход (IE): только сортировка по возрастанию или по убыванию;

Внешний ключ (FK);

Супер-ключ - совокупность всех потенциальных ключей отношения;

Ключ отношения (RK).

**Понятие ключа отношения**

Ключ отношения - это совокупность первичного ключа и всех внешних ключей, образующих идентифицирующие связи.  **RK = PK + {FK}**

**Миграция ключа отношения:**

Ключ отношения родительского отношения = Внешний ключ дочернего отношения!

**Types of keys of the relational data model:**

Potential key;

Primary key;

Alternative key: prohibition of repeated attribute values, sorting in ascending or descending order;

Inversion entry: only sorting in ascending or descending order;

Foreign key;

Super-key - the set of all potential keys of the relation;

Relation key.

**The concept of a relation key**

A relation key is the combination of the primary key and all foreign keys that form identifying relationships.

**Relation key migration:**

Parent relation key = Foreign key of child relation!

**Оптимизация базы данных**

Технический анализ данных об изменении курса национальной валюты производится, например, за истекшие сутки. В таблице, где фиксируются все колебания курса в течение суток, записи с датой необходимой для технического анализа располагаются в конце, так как внесены туда с наибольшим значением ДатаВремя. Для выборки строк для проведения технического анализа придется перебирать все записи, пока не будут найдены искомые. Для уменьшения времени на выполнение данной операции следует пересортировать строки по убыванию значения ДатаВремя. Если допускаются повторяющиеся значения, то нужно использовать Инверсионный вход (IE) с сортировкой по убыванию (DESC). Иначе - альтернативный ключ (АК), также с сортировкой по убыванию. В этом случае искомые данные окажутся вверху таблицы и будут найдены сразу. А поиск будет завершен сразу после того, как будет найдена последняя запись за предыдущие сутки.

**Database optimization**

Technical analysis of data on changes in the exchange rate of the national currency is carried out, for example, over the past day. In the table where all rate fluctuations are fixed during the day, records with the date required for technical analysis are placed at the end, since they are entered there with the highest DateTime value. To select lines for technical analysis, you will have to go through all the records until the desired ones are found. To reduce the time to perform this operation, re-sort the rows in descending order of the DateTime value. If duplicate values are allowed, then the Inversion Entry (IE) sorted in descending order (DESC) should be used. Otherwise - Alternative Key (AK), also sorted in descending order. In this case, the required data will be at the top of the table and will be found immediately. And the search will be completed immediately after the last record for the previous day is found.

Семинар 11 – **Целостность базы данных (на уровне одного, двух и трех отношений)**

**Целостность базы данных**

Уровень отдельного отношения - свойства ограничения целостности отношения:

**C1**. Дублирование строк таблицы не допускается.

**C2**. Порядок строк в отношении не существенен.

**K1**. Каждое отношение имеет, по крайней мере, один ключ (состоящий из всех атрибутов).

**K2**. Значение ключа уникально идентифицирует кортеж отношения (не существует двух строк, которые имели бы равные значения атрибутов, входящих в ключ и рассматриваемых как одно целое).

**K3**. Никакое подмножество атрибутов ключа, которое образуется при удалении из ключа любого атрибута, не обладает свойством уникальности идентификации (свойств), что называется минимальностью ключа.

**K4**. Первичный ключ (один из потенциальных ключей отношения) не допускается обновлять или оставлять без значения.

**K5**. С помощью ключей должны поддерживаться неявные связи между отношениями, отображающие семантику предметной области и обеспечивающие корректность транзакций (изменений в значениях атрибутов).

**A1**. Атрибуты отношений должны быть определены по типу (integer, char, BLOB, real,..) и формату представления.

**A2**. Диапазон области допустимых значений атрибута может быть ограничен с помощью средств языка запросов.

**A3**. При присвоении каждому столбцу уникального имени роли порядок столбцов также становится несущественным.

Уровень связанных таблиц: Правила ссылочной целостности (**R**eferential **I**ntegrity - RI)

**R**ESTRICT – стандартное правило (неудаления) реляционной модели данных

**C**ASCADE – правило каскадного преобразования

**S**ET **D**EFAULT – правило изменения ссылки на предусмотренное значение (по-умолчанию)

**S**ET **N**ULL – аналогичное правилу SET DEFAULT для неидентифицирующей связи

**N**ONE – правило не реагирования на событие

**N**O **A**CTION – отсутствие правила реагирования

**Отдел Сотрудник**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ИД | Наименование | ... |  | ИД (FK) | ИД\_С | Ф.И.О. | Должность | ... |
| 102 | Трейдерский отдел |  |  | 102 | 2307 | Иванов М.Н. | Старший аналитик |  |
| 224 | Отдел теханализа |  |  | 102 | 8932 | Судоргин В.А. | Оператор |  |
| 509 | Бухгалтерия |  |  | 224 | 1193 | Семенецкий М.А. | Старший аналитик |  |
|  |  |  |  | 509 | 6739 | Порошенко П.А. | Главный бухгалтер |  |

**I**nsert : **R**estrict

**U**pdate : **R**estrict

**D**elete : **N**one

**D**elete : **R**estrict

**U**pdate : **R**estrict

**I**nsert : **N**one

Уровень фрагмента структуры из трех связанных таблиц: Ловушки соединения (Connection Trap)

|  |  |  |
| --- | --- | --- |
| **Ловушка разветвления** |  | C:\E32C8A85\662DC750-E7BA-4A89-9A15-A63FDF2017A3.files\image026.png |
| **Ловушка разрыва** |  |
| **НЕТ ЛОВУШКИ!** |  |

**Ловушка разветвления**

Сотрудник 
# Учреждение 
REF (Учреждение. Наименование) 
работает 
Учреждение 
# Наименование 
состоит из 
Отдел 
# Наименование 
# Учреждение 

О труд ние 
Сид ров 
огу 
«Поле » 
Б хг лтерив 

**В каком отделе работает Степанов?**

**What department does Stepanov work in?**

ЛЕЧЕНИЕ:

Сотрудник 
# Учреждение 
REF (Учреждение. Наименование) 
работает 
Учреждение 
# Наименование 
состоит из 
Отдел 
# Наименование 
# Учреждение 

Эквивалентная схема:

Учреждение 
# Наименование 
состоит из 
Отдел 
# Наименование 
# Учреждение 
работает 
Сотрудник 
# Учреждение 
REF (Отдел. Учреждение) 
# Отдел 
REF (Отдел. Наименование) 

Автоматически созданный замещающий текст:



**Ловушка разрыва**

Отдел 
# Наименование 
работает 
Сотрудник 
# Отдел 
REF (Отдел.Наименование) 
под готовил 
Документ 
# Наименование 
# Отдел 
# Сотрудник 

Автоматически созданный замещающий текст:



ЛЕЧЕНИЕ

Отдел 
# Наименование 
работает 
Сотрудник 
# Отдел 
REF (Отдел.Наименование) 
под готовил 
Документ 
# Наименование 
# Отдел 
# Сотрудник 

Эквивалентная схема:

Отдел 
# Наименование 
работает 
Сотрудник 
# Отдел 
REF (Отдел.Наименование) 
под готовил 
Документ 
# Наименование 
# Отдел 
# Сотрудник аЕЕ(Сотрудник.ФИО) 

год “ 
Зу 
алтер 
епаноа 
идороа 
год е 

Задания для решения на семинаре и дома:

Задача 1: Дано:

Врач 
проводИТ Дата проведения 

Разработать правила ссылочной целостности для связи между отношениями.

Задача 2: Дано:

Валюта 
# Код 
Сделка 
участвует в 
Дата проведения 
* Объем 

Разработать правила ссылочной целостности для связи между отношениями.

Задача 3: Дано:

Индекс 
Код 
ведет 
Биржа 
# Код 
торгует 
Акция 
Наименование 
* Вид 

Исследовать и устранить ловушку соединения (connection trap).

Задача 4: Дано:

Сделка 
Дата/Время 
* Объем 
проводит 
Трейдер 
объединяет 
Биржа 
# Код 

Исследовать и устранить ловушку соединения (connection trap).

Семинар 12 – **Подъязык описания данных (DDL) языка структурированных запросов**

**Схема отношения** (relation schema) – это список атрибутов отношения с указанием их свойств.

A **relation schema** is a list of relation attributes with their properties.

SQL (**S**tructured **Q**uery **L**anguage) = DDL (**D**ata **D**efinition **L**anguage) + DML (**D**ata **M**anipulation **L**anguage) + TCL (**T**ransaction **C**ontrol **L**anguage)

*<Имя\_отношения> (<Имя\_атрибута> <Имя\_домена> [<Имя\_ключа> NULL | NOT NULL <Имя\_правила\_валидации> <Имя\_значения\_по\_умолчанию>*]*,*

*<Имя\_атрибута> <Имя\_домена> [<Имя\_ключа> NULL | NOT NULL <Имя\_правила\_валидации> <Имя\_значения\_по\_умолчанию>*]*,…,*

*<Имя\_атрибута> <Имя\_домена> [<Имя\_ключа> NULL | NOT NULL <Имя\_правила\_валидации> <Имя\_значения\_по\_умолчанию>*]);

**CREATE TABLE <Схема\_отношения>;**

**Основы подъязыка DDL**

CREATE – создание объекта базы данных по спецификации в предложении.

ALTER – изменение структуры объекта базы данных в соответствие со спецификацией из SQL-предложения.

DROP – уничтожение объекта базы данных, если это не повлечет нарушение целостности данных.

Создание неиндексированной таблицы

Marka 
# IDM: SMALL INT 
* Producer: VARCHAR2(15) 
* Model: VARCHAR2(15) 
o Country: CHAR(2) 
made in 
A utos 
# Gosn: VARCHAR2(9) 
# IDM: SMALLINT REF (Marka.lDM) 
o Dated: DATE 
o Distance: INTEGER 
o Category: VARCHAR2(50) 
o Price. INTEGER 

**CREATE TABLE Marka (IDM SMALLINT NOT NULL,**

**Producer VARCHAR2(15) NOT NULL,**

**Model VARCHAR2(15) NOT NULL,**

**Country CHAR(2));**

**CREATE TABLE Autos (Gosn VARCHAR2(9) NOT NULL,**

**IDM SMALLINT NOT NULL,**

**DateB DATE,**

**Distance INTEGER,**

**Category VARCHAR2(50),**

**Price INTEGER);**

   Добавление ключа отношения в неиндексированную таблицу

**ALTER TABLE Marka ADD PRIMARY KEY (IDM);**

**ALTER TABLE Autos ADD PRIMARY KEY (IDM, Gosn);**

Создание внешнего ключа идентифицирующей связи

**ALTER TABLE Autos ADD FOREIGN KEY (IDM) REFERENCES Marka(IDM);**

Создание внешнего ключа не идентифицирующей связи

C:\E32C8A85\662DC750-E7BA-4A89-9A15-A63FDF2017A3.files\image050.png

**ALTER TABLE Autos ADD PRIMARY KEY (IDM, Gosn);**

**ALTER TABLE Autos ADD FOREIGN KEY (IDM) REFERENCES Marka(IDM);**

Задание для самостоятельного выполнения

Отдел 
# Наименование 
работает 
Сотрудник 
# Отдел 
REF (Отдел.Наименование) 
под готовил 
Документ 
# Наименование 
# Отдел 
# Сотрудник аЕЕ(Сотрудник.ФИО) 

Переименование таблицы

**ALTER TABLE Autos RENAME TO Vehicle;**

Добавление столбца в таблицу

**ALTER TABLE Autos ADD (Owner CHAR(50) NOT NULL);**

Изменение свойств столбца, например, типа данных

**ALTER TABLE Autos MODIFY Owner VARCHAR2(100);**

Изменение свойств столбца, например, переименование

**ALTER TABLE Autos RENAME Owner TO Owner\_FIO;**

Уничтожение столбца таблицы

**ALTER TABLE Autos DROP COLUMN Owner\_FIO;**

Уничтожение таблицы

**DROP TABLE Autos;**

**DROP TABLE Marka;**

Примечание! Удалить родительскую таблицу можно только после удаления всех дочерних, ссылающихся на нее.

**Задание на дом:** Создать таблицы Marka и Vehicle в соответствие с моделью на рисунке

Marka 
# IDM: SMALL INT 
* Producer: VARCHAR2(15) 
* Model: VARCHAR2(15) 
o Country: CHAR(2) 
made in 
Vehicle 
# Gosn: VARCHAR2(9) 
# IDM: SMALLINT REF (Marka.lDM) 
o Dated: DATE 
o Distance: INTEGER 
o Category: VARCHAR2(50) 
o Price. INTEGER 

Семинар 13 – **Подъязык манипулирования данными (DML) и управления транзакциями (TCL)**



**Подъязык манипулирования данными DML**

SELECT – выборка данных

INSERT – вставка данных

UPDATE – модификация данных

DELETE – удаление данных

MERGE – вставка данных слиянием: если данные существуют, то они обновляются, если не существуют, то вносятся в таблицу

**Понятие транзакции**

**Транзакция** - это процедура перевода базы данных из одного целостного состояния в другое.

Если транзакция началась, то она должна обязательно завершиться, или вернуть базу данных в исходное состояние до начала транзакции.

*Свойства транзакций (ACID-свойства):*

1. **Атомарность** (**A**tomicity, неделимость), означающая, что выполняются либо все процедуры транзакции, либо ни одной. В случае неуспешного выполнения операции транзакции, все уже выполненные операции безусловно откатываются к исходному состоянию базы данных.

2. **Согласованность** (**C**onsistency) означает, в результате выполнения транзакции база данных переходит из исходного согласованного (целостного) состояния в новое состояние, удовлетворяющее требованиям согласованности и целостности.

3. **Сериализуемость** (**I**nsolation, изолированность), означающая, что разные транзакции не оказывают неожиданного воздействия друг на друга и результат совместного выполнения нескольких транзакций эквивалентен результату их последовательного выполнения.

4. **Сохранность** (**D**urability, сохраняемость), означающая, что если транзакция зафиксирована, то результат ее выполнения не пропадает даже в случае сбоя вычислительной системы сервера.

ВЫБОРКА данных не является транзакцией!

ВСТАВКА, УДАЛЕНИЕ, МОДИФИКАЦИЯ, СЛИЯНИЕ - транзакции!

Все преобразования, предусмотренные транзакцией, производятся над копией данных, помещенной в оперативной памяти сервера баз данных. Для этого первой операцией транзакции является выборка строк в оперативную память. После преобразования данных производится их ФИКСАЦИЯ - запись результата транзакции в файл данных на накопителе сервера баз данных. Сбой на любом этапе транзакции ОДНОЗНАЧНО приводит в действие механизм отката транзакции, который из СЕГМЕНТА ОТКАТА возвращает данные в файл данных. Смотри мультик:



**Блокировки**

**Блокировка** (англ. *lock*) в СУБД — отметка о захвате объекта базы данных транзакцией в ограниченный или исключительный доступ с целью предотвращения коллизий и поддержания целостности **данных**.

Типы блокировок:

1. ***Shared*** Lock - разделяемая *блокировка*, которая используется при выполнении операции *чтения данных*. Позволяется *чтение данных* другой транзакцией, но запрещено изменение данных.
2. ***Exclusive*** *Lock* - монопольная *блокировка*, которая применяется при изменении данных. Эта *блокировка* полностью запрещает *доступ к данных* другим транзакциям.
3. ***Update*** Lock - *блокировка обновления*, которая является промежуточной между разделяемой и *монопольной блокировкой*. Используется, когда *транзакция* хочет обновить данные в какой-то ближайший момент времени, но не сейчас, и, когда этот момент придет, не хочет ожидать другой *транзакции*. В этом случае другим транзакциям разрешается устанавливать *разделяемые блокировки*, но не позволяет устанавливать монопольные.

*Блокировки* могут устанавливаться на трех уровнях -

Уровень строк - уровень по умолчанию.

Уровень страницы - используется для более эффективного использования ресурсов и в случае если в данной странице много строк с этой блокировкой.

Уровень таблицы - используется при операциях над таблицей и в случае эскалации блокировок.

**Язык управления транзакциями (TCL)**

COMMIT – принудительная фиксация транзакции

ROLLBACK – принудительный откат последней не зафиксированной (COMMIT) транзакции

ROLLBACK ALL – принудительный откат всех транзакций, которые были выполнены после последней фиксации (COMMIT) или отката (ROLLBACK)

ROLLBACK TO <Имя\_метки> - принудительный откат всех транзакций, которые были выполнены после объявления метки с <Имя\_метки>

SAVEPOINT <Имя\_метки> - создание метки с именем метки, после чего становится возможным откат всех транзакций, выполненных после нее

**Вставка строк (INSERT)**

Marka 
# IDM: SMALL INT 
* Producer: VARCHAR2(15) 
* Model: VARCHAR2(15) 
o Country: CHAR(2) 
made in 
Vehicle 
# Gosn: VARCHAR2(9) 
# IDM: SMALLINT REF (Marka.lDM) 
o Dated: DATE 
o Distance: INTEGER 
o Category: VARCHAR2(50) 
o Price. INTEGER 

**INSERT INTO <Упрощенная схема отношения> VALUES (<Данные в порядке столбцов, определенных в упрощенной схеме отношения>);**

**INSERT INTO Marka(IDM, Producer, Model, Country) VALUES(<Данные для IDM>, <Данные для Producer>, <Данные для Model>, <Данные для Country>);**

**INSERT INTO Marka(IDM, Producer, Model, Country) VALUES(1, 'Mercedes-Benz', 'S600', 'DE');**

**INSERT INTO Marka(IDM, Producer, Model, Country) VALUES(2, 'Audi', 'A8', NULL);**

эквивалентное SQL-предложение

**INSERT INTO Marka(IDM, Producer, Model) VALUES(2, 'Audi', 'A8');**

вставка без указания схемы отношения, но в точном соответствие со спецификацией словаря данных

**INSERT INTO Marka VALUES(3, 'Audi', 'A6', 'DE');**

переопределение схемы отношения

**INSERT INTO Marka(Country, Model, Producer, IDM) VALUES('JP', 'LX300', 'Lexus', 4);**

вставка в дочернюю таблицу только после вставки в родительскую (правило ссылочной целостности RESTRICT)!

**INSERT INTO Vehicle(IDM, Gosn, DateB, Distance, Category, Price)**

**VALUES(3, 'y309ak777', to\_date('25.03.2021', 'dd.mm.yyyy'), 28934, 'VIP-Такси', 3450000);**

вставка значения, получаемого с помощью вложенного запроса - подзапроса (он должен возвращать унарное значение - одна строка в одном столбце)

**INSERT INTO Vehicle(IDM, Gosn, DateB, Distance, Category, Price)**

**VALUES(3, 'y312ak777', (SELECT sysdate FROM dual), 12, 'VIP-Такси', 4500000);**

**INSERT INTO Vehicle(IDM, Gosn, DateB, Distance, Category, Price)**

**VALUES((SELECT IDM FROM Marka WHERE Producer = 'Audi' AND Model = 'A6'),**

**'y326ak777', to\_date('03.04.2021', 'dd.mm.yyyy'), 23306, 'VIP-Такси', 3450000);**

вставка выбранных из таблицы значений в другую таблицу или переменную

**SELECT <Список\_столбцов> INTO <Имя\_таблицы\_в\_которую\_вставляются\_данные>**

**FROM <Имя\_таблицы\_из\_которой\_производится\_выборка>**

**WHERE <Условия\_отбора\_строк>;**

Примечание! Таблица, в которую вставляются данные, будет создана автоматически на основании <Списка\_столбцов>. Если SQL-предложение применяется в процедуре или функции, где была создана переменная с именем <Имя\_таблицы\_в\_которую\_вставляются\_данные>, то в случае совпадения ее структуры с отобранным кортежем, он будет занесен в нее.

**ЗАДАНИЕ НА ДОМ:**

Сформировать SQL-предложения и вставить в таблицу Marka не менее 8 строк с наименованиями производителей и моделей автомобилей. Сформировать и вставить в таблицу Vehicle не менее 2 автомобилей для каждой модели, зарегистрированной в таблице Marka! В таблице Vehicle должно быть не менее 16 строк.

SQL-предложения записать на странице "Домашняя работа", а в APEX выполнить SQL-транзакции в соответствие с разработанными SQL-предложениями.

Семинар 14 – **Модификация и удаление данных**

**Модификация данных (UPDATE)**

Модификация данных - запись нового значения в указанный атрибут существующей строки (в отличие от вставки, когда значения вносятся целой, новой для таблицы, строкой.

**UPDATE <Имя таблицы> SET <Имя столбца> = <Вносимое значение>, <Имя столбца> = <Вносимое значение>, ... [WHERE <Условие отбора>];**

**<Имя таблицы>** указывает, в какой таблице необходимо изменить данные.

**<Имя столбца> = <Вносимое значение>** специфицирует собственно изменение и может предписывать одновременную модификацию сразу в нескольких столбцах.

**<Условие отбора>** специфицирует набор строк (для которых оно истинно), в которых надлежит произвести изменения. Если условие отбора не указано, то изменения производятся во всех строках таблицы!

УПРАЖНЕНИЯ:

1. Зарегистрировать автомобиль KIA K5, государственный номер y330kx199, дата приобретения 22 ноября 2019 года. Пробег, стоимость и категория не известны.

**INSERT INTO Marka(IDM, Producer, Model, Country) VALUES(8, 'KIA', 'K5', 'KR'); -- вставка записи о модели**

**INSERT INTO Vehicle(IDM, Gosn, DateB, Distance, Category, Price)**

**VALUES(8, 'y330kx199', to\_date('22.11.2019', 'dd.mm.yyyy'), NULL, NULL, NULL); -- вставка записи об автомобиле**

1. Внести в базу данных цену приобретения автомобиля в размере 1300000 руб.

**UPDATE Vehicle SET Price = 1300000 WHERE Gosn = 'y330kx199';**

1. Внести в базу данных пробег автомобиля 108906 км и его категорию - "Поручения"

**UPDATE Vehicle SET Distance = 108906, Category = 'Поручения' WHERE Gosn = 'y330kx199';**

1. Изменить пробег автомобиля на новое значение 114591 км

**UPDATE Vehicle SET Distance = 114591 WHERE Gosn = 'y330kx199';**

ПРИМЕЧАНИЕ: Результат выполнения SQL-предложения можно увидеть, послав в базу данных команду на выборку. Например, **SELECT \* FROM Vehicle;**

**Условие отбора строк для модификации (WHERE)**

Секция WHERE содержит логические условия отбора строк для выполнения предписанного действия (UPDATE, SELECT, DELETE). Если для строки логическое условие истинно, то она включается в множество, в котором производится преобразование. Если условие ложно, то строка не используется.

Условие отбора строк вычисляется для каждой строки таблицы. Если условие отбора строк не задано, то для преобразования отбираются все строки таблицы.

Условий отбора в секции WHERE может быть несколько. При этом они организуются в логические выражения с помощью предикатов AND, OR, NOT, а также операторных скобок, определяющих порядок вычисления выражения.

**ПРЕДУПРЕЖДЕНИЕ!** Отсутствие секции WHERE приводит к преобразованию (UPDATE, DELETE) всех, без исключения, строк таблицы!

УПРАЖНЕНИЯ:

1. Изменить категорию всех автомобилей, пробег которых превышает 100000 км, на "Резерв"

**UPDATE Vehicle SET Category = 'Резерв' WHERE Distance >= 100000;**

1. Изменить наименование производителя всех автомобилей Lexus на новое наименование производителя "Toyota-Lexus"

**UPDATE Marka SET Producer = 'Toyota-Lexus' WHERE Producer = 'Toyota';**

1. Изменить категорию всех автомобилей из категории "Резерв" на категорию "Списание", если их цена не превышает 1500000 рублей

**UPDATE Vehicle SET Category = 'Списание' WHERE Price <= 1500000 AND Category = 'Резерв';**

1. Автомобиль с государственным номером y631kb790 перерегистрирован в ГИБДД и получил государственный номер x703cc799

**UPDATE Vehicle SET Gosn = 'x703cc799' WHERE Gosn = 'y631kb790';**

1. Увеличить пробег автомобилей с государственными номерами c501ac799, c290ax797 и y659kb790 на 115 км

**UPDATE Vehicle SET Distance = Distance + 115 WHERE Gosn = 'c501ac799' OR Gosn = 'c290ax797' OR Gosn = 'y659kb790';**

Решение этой же задачи с применением предиката вхождения в список значений IN

**UPDATE Vehicle SET Distance = Distance + 115 WHERE Gosn IN ('c501ac799', 'c290ax797', 'y659kb790');**

1. Изменить категорию автомобилей из категории "Резерв" и имеющих пробег более чем 90000 км, на категорию "Модернизация"

**UPDATE Vehicle SET Category = 'Списание' WHERE Category = 'Резерв' AND Distance > 90000;**

**Удаление данных (DELETE)**

Удаление данных - это запись в ячейку таблицы значения NULL. Если необходимо удалить часть строки (одну или несколько, но не все ячейки), то используется предложение UPDATE, а в секции SET указывается целевое значение NULL (строка остается, изменяется только ее часть). Если необходимо удалить строку (во все ячейки записать значение NULL), то применяется SQL-предложение DELETE.

**ПРИМЕЧАНИЕ!** Только в СУБД ORACLE для очистки таблицы от данных - удаления всех строк сразу, можно использовать предложение **TRUNCATE TABLE <Имя таблицы>;**

**DELETE FROM <Имя таблицы> [WHERE <Условие отбора>];**

**ПРЕДУПРЕЖДЕНИЕ!** Отсутствие секции WHERE приводит к удалению всех, без исключения, строк таблицы!

УПРАЖНЕНИЯ:

1. Отключить в СУБД функцию автоматической фиксации транзакции AUTOCOMMIT. Для этого в APEX сбросить флаг опции Autocommit в левой верхней части формы Автоматически созданный замещающий текст:

   . В PostgreSQL, инструмент Query Tool, найти в панели кнопок конструкцию запуска на исполнение запроса, как на картинке Автоматически созданный замещающий текст:

   . Нажать на кнопку "Стрелка вниз" Автоматически созданный замещающий текст:

    и отменить действие AUTOCOMMIT.

1. Удалить запись об автомобиле с государственным номером y631kb790

**DELETE FROM Vehicle WHERE Gosn = 'y631kb790';**

1. Откатить транзакцию

**ROLLBACK;**

1. Удалить данные обо всех автомобилях

**DELETE FROM Vehicle;**

Для СУБД ORACLE (выполняется в APEX)

**TRUNCATE TABLE Vehicle;**

1. Откатить транзакцию

**ROLLBACK;**

1. Включить функцию AUTOCOMMIT

**Задание на дом:**

1. Разработать SQL-предложение для изменения наименования категории "VIP-такси" на "Заказная".
2. Разработать SQL-предложение для изменения стоимости всех автомобилей на 15%.
3. Разработать SQL-предложение для удаления автомобилей категории "Списание".
4. Разработать SQL-предложение для удаления автомобилей, пробег которых превышает 300000 км.

Семинар 15 – **Выборка данных. Первое знакомство с оператором SELECT**

**Простая выборка данных (SELECT)**

**SELECT <Список столбцов выдачи и их преобразований>**

**FROM <Список таблиц, из которых следует выбрать данные>**

**[WHERE <Условия отбора записей>]**

**[GROUP BY <Столбцы, по котором следует провести группирование>]**

**[HAVING <Условия отбора сгруппированных строк>]**

**[ORDER BY <Список столбцов в порядке сортировки> ASC | DESC]**

**[START WITH <Условие для начала обхода иерархии - входящая вершина>] -- Только для СУБД ORACLE!**

**[CONNECT BY <Условие для окончания обхода иерархии - завершающая вершина поддерева>] -- Только для СУБД ORACLE!**

**[FOR UPDATE] -- Блокировка типа UPDATE**

**[NO WAIT]; -- Не ждать разблокировки отдельных строк**

УПРАЖНЕНИЯ:

1. Составить список моделей автомобилей.

**SELECT Model FROM Marka;**

1. Составить список государственных номерных знаков и пробега автомобилей.

**SELECT Gosn, Distance FROM Vehicle;**

**Управление набором столбцов**

Перечень столбцов выдачи указывается в секции SELECT в произвольном порядке. Если необходимо в выдачу включить все столбцы и порядок их следования не важен (как в схеме отношения при создании таблицы), то вместо их списка можно указать символ "\*" (звездочка).

УПРАЖНЕНИЯ:

1. Вывести содержимое таблицы Marka.

**SELECT \* FROM Marka;**

1. Отобразить список всех автомобилей, зарегистрированных в базе данных.

**SELECT \* FROM Vehicle;**

1. Отобразить перечень стран и производителей автомобилей.

**SELECT Country, Producer FROM Marka;**

1. Показать государственные номера, даты приобретения и пробеги зарегистрированных автомобилей.

**SELECT Gosn, DateB, Distance FROM Vehicle;**

**Подавление повторяющихся значений (DISTINCT)**

Для исключения повторяющихся значений перед именем столбца следует применить ключевое слово DISTINCT. Его область действия распространяется на всю комбинацию столбцов в секции SELECT. Ключевое слово DISTINCT должно следовать только после ключевого слова SELECT.

УПРАЖНЕНИЯ:

1. Выдать список производителей без повторений.

**SELECT DISTINCT Producer FROM Marka;**

1. Выдать перечень категорий, исключая повторяющиеся значения.

**SELECT DISTINCT Category FROM Vehicle;**

ПРИМЕЧАНИЕ: Значение NULL обрабатывается, как полноправное значение, хранящееся в таблице.

**Псевдоним столбца, псевдоним таблицы**

Каждому столбцу в зоне действия запроса может быть присвоен псевдоним, который прекращает свое существование сразу после завершения выполнения запроса.

**SELECT <Имя столбца> [AS] "<Псевдоним>" FROM <Таблица>…;**

Примечание: Действующая редакция стандарта SQL разрешает опустить ключевое слово AS и кавычки. В случае применения псевдонима на кириллице кавычки опускать не следует.

В рамках запроса по аналогичным правилам могут быть применены псевдонимы для неограниченного подмножества столбцов, вплоть до всех.

**SELECT <Имя столбца> [AS] "<Псевдоним>", <Имя столбца> [AS] "<Псевдоним>",… , <Имя столбца> [AS] "<Псевдоним>" FROM <Таблица>…;**

Для соединения таблиц и в других целях в рамках запроса могут применяться псевдонимы таблиц. Правила их определения такие же, как и для псевдонимов столбцов.

**SELECT … FROM <Имя таблицы> [] "<Псевдоним>" …;**

УПРАЖНЕНИЯ:

1. Сформировать таблицу моделей автомобилей, указав имена столбцов выдачи "Бренд" - Producer, "Модель" - Model, "Страна" - Country.

**SELECT Producer AS "Бренд", Model AS "Модель", Country AS "Страна" FROM Marka;**

1. Сформировать ведомость приобретения автомобилей, где указать государственный номерной знак, дату приобретения и стоимость каждого автомобиля.

**SELECT Gosn AS "Государственный номерной знак",**

**DateB AS "Дата приобретения",**

**Price AS "Стоимость приобретения"**

**FROM Vehicle;**

1. Сформировать список моделей автомобилей, определив псевдоним для таблицы, как "Модели автомобилей".

**SELECT Model AS "Модели автомобилей" FROM Marka AS "Модели";**

**Контейнерная нотация имени столбца**

Для обеспечения однозначности восприятия объекта базы данных (столбца, таблицы, индекса, представления и тому подобного) в запросе применяется контейнерная (точечная) нотация его имени. Формат контейнерной нотации: **[<Имя суперконтейнера>.]<Имя контейнера>.<Имя объекта>**.

Здесь в качестве суперконтейнера подразумевается более высокий уровень обобщения, например, для описания столбца может быть указано имя схемы пользователя (имя учетной записи владельца объекта). Примеры: **public.marka -- таблица Marka схемы public**

**public.marka.producer -- столбец Producer таблицы Marka схемы public**

**public.vehicle.price -- столбец Price таблицы Vehicle схемы public**

**marka.idm -- столбец IDM таблицы Marka**

**vehicle.IDM -- столбец IDM таблицы Vehicle**

УПРАЖНЕНИЯ:

1. Сравните результаты двух запросов: **SELECT Country, Model FROM Marka;**

и **SELECT Marka.Country, Marka.Model FROM Marka;**

1. Из какой таблицы должны быть извлечены данные для конструкции **Vehicle.DateB, Vehicle.IDM, Marka.IDM, public.Marka.Country, scott.Marka.IDM?**
2. В чем отличие между запросами **SELECT Marka.IDM FROM Marka;**

и **SELECT Vehicle.IDM FROM Vehicle;**

**Задание на дом:**

1. Сформировать список дат, в которые приобретались зарегистрированные автомобили.
2. Составить список стран, бренды которых представлены зарегистрированными автомобилями.
3. Составить список государственных номерных знаков, пробега и принадлежности к категории зарегистрированных автомобилей (применить псевдонимы столбцов).
4. Во всех запросах использовать контейнерную нотацию для именования столбцов.

Семинар 16 – **Представление**



**Представление**

Представление (англ. - VIEW) - это объект базы данных, представляющий собой спецификацию для оперативной выборки данных по предусмотренным заранее критериям и условиям.

Представление хранится в базе данных в виде SQL-предложения, обладающего уникальным именем. Оно автоматически выполняется и создает временную таблицу, наполненную данными выборки, размещенную на все время сеанса пользователя в памяти сервера баз данных.

Для обеспечения информационных потребностей пользователя можно создать любое необходимое количество представлений любой структуры. В рамках представления таблицы базы данных могут быть соединены между собой, могут быть произведены расчеты и вычислены значения функций.

Работа с представлением никак не влияет на данные, хранящиеся в таблицах.

Через представление, основанное на единственной таблице, доступно выполнение транзакций. Если представление основано на нескольких таблицах, то оно не позволяет выполнять транзакции.

A view is a database object, which is a specification for an operational data retrieval according to predetermined criteria and conditions. The view is stored in the database as an SQL statement with a unique name. It is automatically executed and creates a temporary table filled with sample data, placed for the duration of the user session in the memory of the database server. To meet the information needs of the user, you can create any necessary number of views of any structure. Within a view, database tables can be interconnected, calculations can be made, and function values calculated. Working with the view does not affect the data stored in the tables in any way. Through a view based on a single table, transaction execution is available. If a view is based on multiple tables, then it does not allow transactions.

**Материализованное представление**

Материализованное представление (англ. - MATERIALIZED VIEW) - это разновидность представлений (VIEW), которые в промежутках между сеансами пользователя хранят выборку в файле данных. Они, по сути, являются специально преобразованными на основе данных из таблиц объектами базы данных с аналогичным таблицам поведением.

Для актуализации данных, размещенных в представлении, необходимо выполнить специальное SQL-предложение REFRESH. Тогда, как в представлении (VIEW) данные обновляются всякий раз, когда пользователь обращается к нему.

**ПРИМЕЧАНИЕ!** Вставить данные через материализованное представление не возможно. Данные могут быть вставлены в родительскую таблицу только через представление (не материализованное - VIEW) или непосредственно с помощью SQL-предложения INSERT.

A materialized view is a type of view that stores a selection in a data file between user sessions. They, in fact, are database objects specially converted based on data from tables with behavior similar to tables. To update the data placed in the view, it is necessary to execute a special REFRESH SQL statement. Whereas in the view the data is updated whenever the user accesses it. NOTE! Inserting data through a materialized view is not possible. Data can only be inserted into the parent table through a view (not materialized) or directly using an SQL INSERT statement.

**Работа с представлением**

Создание или преобразование структуры представления:

**CREATE [OR REPLACE] VIEW <Имя представления> AS <Запрос на выборку данных>;**

УПРАЖНЕНИЯ:

1. Создать представление с именем "Автомобили" для чтения списка всех зарегистрированных автомобилей, указав их государственный регистрационный знак и дату приобретения.

**CREATE OR REPLACE VIEW "Автомобили" AS**

**SELECT Gosn AS "Государственный регистрационный знак",**

**to\_char(DateB, 'dd.mm.yyyy') AS "Дата приобретения"**

**FROM Vehicle;**

1. Создать представление с именем "Модели автомобилей" для просмотра всех моделей автомобилей по каждому бренду.

**CREATE OR REPLACE VIEW "Модели автомобилей" AS**

**SELECT Producer AS "Производитель",**

**Model AS "Модель"**

**FROM Marka;**

1. Создать представление с именем "Вставка модели" для ввода новых моделей автомобилей, предусмотрев следующий порядок ввода: Страна, Производитель, Модель, Идентификатор.

**CREATE OR REPLACE VIEW "Вставка модели" AS**

**SELECT Country AS "Страна",**

**Producer AS "Производитель",**

**Model AS "Модель",**

**IDM AS "Идентификатор"**

**FROM Marka;**

1. Создать представление с именем "Приемка автомобиля" для диспетчера-приемщика, обеспечив ввод им государственного регистрационного знака, идентификатора марки, даты приобретения в формате "день.месяц.год" и пробега принимаемого автомобиля.

**CREATE OR REPLACE VIEW "Приемка автомобиля" AS**

**SELECT Gosn AS "Государственный регистрационный знак",**

**IDM AS "Идентификатор марки",**

**DateB AS "Дата приобретения",**

**Distance AS "Пробег"**

**FROM Vehicle;**

**Выборка данных из представления:**

**SELECT … FROM <Имя представления> [WHERE <Условия отбора строк>];**

УПРАЖНЕНИЯ:

1. Получить из представления "Модели автомобилей" список моделей, производимых в России.
2. Выбрать из представления "Модели автомобилей" все бренды Германии.
3. Сформировать список моделей компании "АвтоВАЗ". Использовать представление "Модели автомобилей".
4. Сформировать список государственных регистрационных знаков автомобилей, имеющих пробег, не превышающий 50000 км. Список получить через представление "Автомобили".
5. Составить список зарегистрированных в базе данных автомобилей, прошедших приемку диспетчером-приемщиком. Указать их государственные регистрационные знаки, даты приобретения в формате "ГОД, ЧИСЛО, МЕСЯЦ", стоимость и пробег.

Вставка строк в таблицу через представление:

**INSERT INTO <Имя представления> (<Список колонок>) VALUES (<Данные строки>);**

УПРАЖНЕНИЯ:

1. Вставить модель "Лада-Веста" производителя "АвтоВАЗ" с идентификатором 100 через представление "Вставка модели".

**INSERT INTO "Вставка модели" ("Страна", "Производитель", "Модель", "Идентификатор")**

**VALUES ('RU', 'АвтоВАЗ', 'Лада-Веста', 100);**

1. Вставить с использованием представления "Приемка автомобиля" машину "Лада-Веста" (Идентификатор марки = 100), государственный регистрационный знак "a631kk790", дата приобретения 21 мая 2021 года, пробег 13568 км.

**INSERT INTO "Приемка автомобиля" ("Идентификатор марки",**

**"Государственный регистрационный знак",**

**"Дата приобретения",**

**"Пробег")**

**VALUES (100, 'a631kk790', to\_date('21.05.2021', 'dd.mm.yyyy'), 13568);**

1. Вставить автомобиль "Лада-Веста" с государственным регистрационным знаком "x902aa777", приобретенную 25 марта 2020 года.
2. Вставить автомобиль "Нива-Легенда" с государственным номерным знаком "o200ao797", приобретенную 6 июля 2019 года и пробежавшую 29360 км.

Модификация данных в таблице через представление:

**UPDATE <Имя представления> SET <Имя столбца> = <Новое значение>**

**[, <Имя столбца> = <Новое значение>]**

**[WHERE <Условия отбора строк>];**

УПРАЖНЕНИЯ:

1. Внести через представление "Приемка автомобиля" пробег автомобиля "x902aa777", составляющий 28564 км.
2. Скорректировать через представление "Приемка автомобиля" пробег автомобиля "a631kk790", увеличив его на 136 км.
3. Через представление "Приемка автомобиля" изменить дату приобретения автомобиля с государственным регистрационным знаком "a631kk790" после его перепродажи, состоявшейся сегодня.
4. Через представление "Приемка автомобиля" исправить ошибки, допущенные диспетчером-приемщиком при приемке автомобиля с государственным регистрационным знаком "a631kk790". А именно, государственный регистрационный знак "a631kс797", пробег составляет 83269 км.

Удаление строк в таблице через представление:

**DELETE FROM <Имя представления> [WHERE <Условия отбора строк>];**

УПРАЖНЕНИЯ:

1. Удалить через представление "Приемка автомобиля" автомобиль с государственным регистрационным знаком "a631kk790".
2. Удалить через представление "Приемка автомобиля" автомобиль с государственным регистрационным знаком "x902aa777".
3. Удалить через представление "Вставка модели" модель "Лада-Веста".
4. Удалить через представление "Вставка модели" модель с идентификатором 101.

Создание материализованного представления:

**CREATE MATERIALAZED VIEW <Имя представления> AS <Запрос на выборку данных>;**

**ПРИМЕЧАНИЕ!** Материализованное представление не может быть реструктурировано (REPLACE) так как хранит данные сразу после его создания.

УПРАЖНЕНИЯ:

1. Создать материализованное представление "Модели гаража", куда включить наименования производителей и моделей из таблицы Marka.
2. Создать материализованное представление "Раритеты" со списком автомобилей (Государственный регистрационный знак, Стоимость, Пробег, Дата приобретения), которые имеют пробег свыше 150000 км и стоят более 3000000 рублей.

Обновление (актуализация) данных, помещенных в материализованное представление:

**REFRESH MATERIALIZED VIEW <Имя материализованного представления>;**

УПРАЖНЕНИЯ:

1. Сформировать список рариретов гаража.
2. Внести в реестр автомобилей через представление "Приемка автомобиля" экземпляр с государственным регистрационным знаком "x561ac177", модели LX200 марки Lexus, дата приобретения 20 сентября 2012 года и пробег 268933 км.
3. Сформировать через представление "Раритеты" список автомобилей с пробегом, превышающим 250000 км. Отыскать среди записей автомобиль с государственным номерным знаком "x561ac177". ПРИМЕЧАНИЕ! Его нет в списке, так как не было выполнено обновление материализованного представления.
4. Произвести обновление материализованного представления Раритеты.

**REFRESH MATERIALIZED VIEW "Раритеты";**

1. Сформировать через представление "Раритеты" список автомобилей с пробегом, превышающим 250000 км. Отыскать среди записей автомобиль с государственным номерным знаком "x561ac177".

Удаление представления (материализованного представления):

**DROP [MATERIALIZED] VIEW <Имя представления / материализованного представления>;**

УПРАЖНЕНИЯ:

1. Удалить представление "Приемка автомобиля".
2. Удалить материализованное представление "Раритеты".

Семинар 17 – **Зачет**

**Вопросы для подготовки к зачету:**

1. Принцип формирования отношения реляционной модели данных.

2. Методология информационного моделирования предметной области «сущность-связь» П. Чена.

3. Методика связывания отношений в информационной (инфологической) модели предметной области.

4. Нотация (графический язык) Баркера для документирования информационной (инфологической) модели предметной области базы данных.

5. Методика разрешения связей типа «многие-ко-многим» в информационной (инфологической) модели предметной области базы данных.

6. Характеристика первой нормальной формы (1НФ) реляционного отношения.

7. Характеристика второй нормальной формы (2НФ) реляционного отношения.

8. Характеристика третьей нормальной формы (3НФ) реляционного отношения.

9. Характеристика нормальной формы Бойса-Кодда (НФБК) реляционного отношения.

10. Характеристика четвертой нормальной формы (4НФ) реляционного отношения.

11. Характеристика пятой нормальной формы (5НФ) – проекционно-соединительной формы (ПСНФ) реляционного отношения.

12. Характеристика доменно-ключевой нормальной формы (ДКНФ) реляционного отношения.

13. Ключи отношений реляционной базы данных.

14. Абстрагирование предметов, процессов и явлений реального мира и их значимых свойств в информационной (инфологической) модели предметной области базы данных.

15. Процедура миграции ключа отношения реляционной базы данных.

16. Домены реляционной модели данных.

17. Абстрагирование связей между предметами, процессами и явлениями реального мира. Виды связей отношений.

18. Отображение логической модели базы данных в ее физическую структуру.

19. Характеристика подъязыков языка структурированных запросов SQL.

20. Характеристика и виды ловушек соединения (connection trap) отношений реляционной базы данных.

21. Моделирование и устранение ловушки разветвления отношений реляционной базы данных.

22. Моделирование и устранение ловушки разрыва отношений реляционной базы данных.

23. Характеристика декомпозиции отношения с соединением без потерь. Принцип соединения кортежей реляционных отношений.

24. Свойства ограничения целостности реляционного отношения.

25. Избыточность в реляционном отношении. Аномалии вставки и способ их устранения.

26. Понятие ссылочной целостности реляционных отношений. Виды правил ссылочной целостности. Главное правило ссылочной целостности реляционной базы данных.

27. Характеристика видов правил ссылочной целостности реляционных отношений.

28. Операторы подъязыка Data Definition Language (DDL) языка структурированных запросов.

29. Схема отношения. Правила составления схемы отношения. Схема базы данных.

30. Понятие «модель данных». Основные положения реляционной модели данных.

31. Свойства ограничения целостности реляционной модели данных.

32. Принцип соединения кортежей реляционного отношения. Условие соединения.

33. Ссылочная целостность. Правила ссылочной целостности.

34. Транзакция. SQL-предложение. Целостное состояние базы данных.

35. Блокировка таблиц. Уровни блокировки. Самоблокировка.

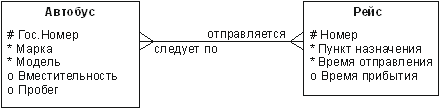
36. Понятие «База данных». Главная цель функционирования базы данных. Основные свойства базы данных.

**Типы задач к зачету:**

1. Дано: Производится учет метеообстановки в районе производственных мощностей предприятия. Ежедневно учитываются: температура воздуха, атмосферное давление, направление и скорость ветра, вид и сила осадков, облачность. Параметры метеообстановки измеряются в нескольких зонах территории промышленной зоны одновременно.

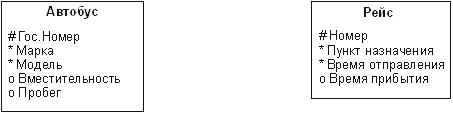
Построить информационную (инфологическую) модель предметной области базы данных.

2. Дано:



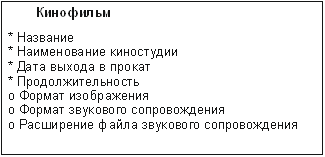
Выполнить разрешение связи типа «многие-ко-многим».

3. Дано:



Связать отношения в инфологической модели.

4. Дано:



Привести отношение ко второй нормальной форме.

5. Дано: отношение *Библиотека (Название\_книги, ФИО\_автора,*

*Город\_издательства, Издательство,*

*Год\_издания, Номер\_экземпляра,*

*Кабинет\_хранения, Стеллаж\_хранения,*

*Полка\_хранения, Состояние\_экземпляра,*

*Дата\_последней\_реставрации\_экземпляра,*

*Регистрационный\_номер\_издания);*

Привести отношение к третьей нормальной форме.

6. Дано: Схема отношения: *Агент (Наименование СТРОКА,*

*Сокращенное наименование СТРОКА,*

*ФИО директора СТРОКА,*

*Юридический адрес СТРОКА,*

*Фактический адрес СТРОКА,*

*Код вида деятельности по ОКВЭД СТРОКА,*

*Дата учреждения ДАТА,*

*Форма собственности СТРОКА,*

*Учредительный капитал ЧИСЛО);*

Привести отношение к нормальной форме Бойса-Кодда, определить свойство NULL/NOT NULL.

7. Дано: отношение *Библиотека (Название\_книги, ФИО\_автора,*

*Город\_издательства, Издательство,*

*Год\_издания, Номер\_экземпляра,*

*Кабинет\_хранения, Стеллаж\_хранения,*

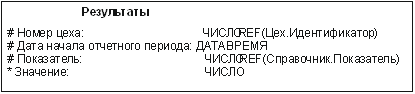
*Полка\_хранения, Состояние\_экземпляра,*

*Дата\_последней\_реставрации\_экземпляра,*

*Регистрационный\_номер\_издания);*

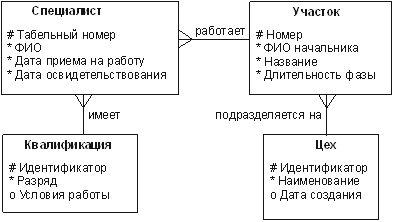
Выполнить нормализацию отношения до ДКНФ. Результат представить в виде совокупности схем отношений.

8. Дано:



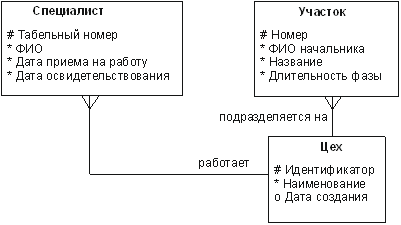
Произвести оптимизацию отношения по времени подведения итогов работы цеха.

9. Дано:



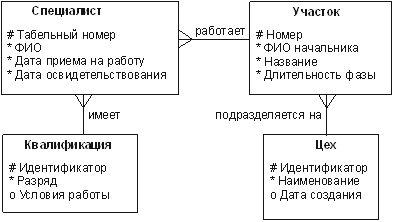
Произвести миграцию ключа отношения.

10. Дано:



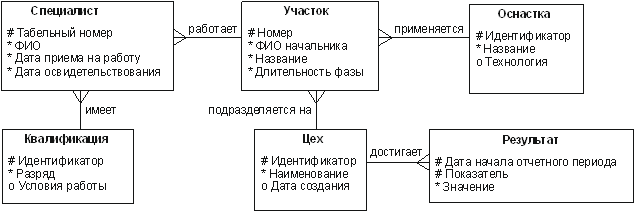
Устранить ловушку соединения.

11. Дано:



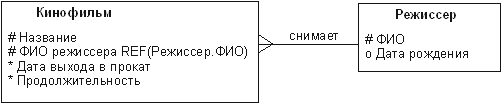
Устранить ловушку разрыва.

12. Дано:



Определить все фрагменты модели с подозрением на ловушки соединения.

13. Дано:



Доказать, что в базе данных отсутствует явление декомпозиции отношения с соединением с потерями.

14. Дано: *Трейдер (Наименование СТРОКА NOT NULL,*

*Юридический адрес СТРОКА NOT NULL,*

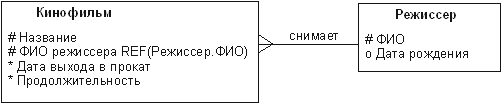
*ФИО директора СТРОКА,*

*Дата учреждения ДАТА NOT NULL,*

*Уставной капитал ЧИСЛО NOT NULL);*

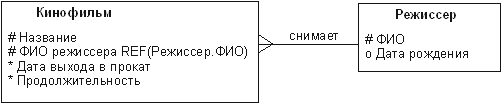
Доказать, что отношение свободно от аномалий вставки.

15. Дано:



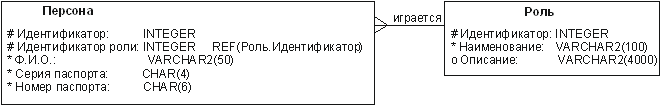
Определить правила ссылочной целостности.

16. Дано:



Сформировать SQL-предложение создания неиндексированной таблицы.

17. Дано:



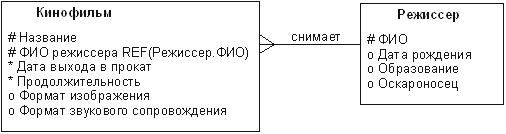
Сформировать SQL-предложения создания первичных и внешних ключей отношений.

18. Дано: CREATE TABLE trand (id smallint NOT NULL,

indicator VARCHAR2(8));

Сформировать SQL-предложение добавления, а затем исключения (второе SQL-предложение) обязательного столбца date\_note, определенного на домене DATETIME в отношение.

19. Дано:



Доказать, что база данных неизбыточна.

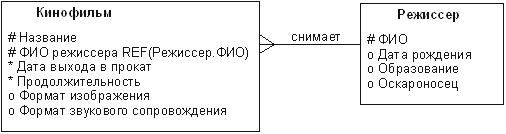
20. Дано: Множества *А=*{*011, 029, 308*}, *В=*{*Волга, Днепр*}, *С=*{*А02, А03, А72*}. Сформировать декартово (прямое) произведение

C:\E32C8A85\662DC750-E7BA-4A89-9A15-A63FDF2017A3.files\image065.png

21. Дано: Радиостанция "Европа+" имеет сеть филиалов по всему миру. Каждый филиал расположен по определенному почтовому адресу, имеет музыкальные студии определенного типа и библиотеку музыкальных записей на различных носителях.

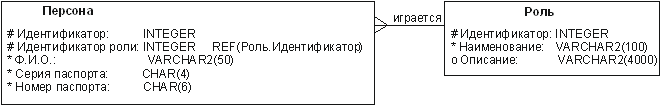
Разработать информационную (инфологическую) модель предметной области.

22. Дано:



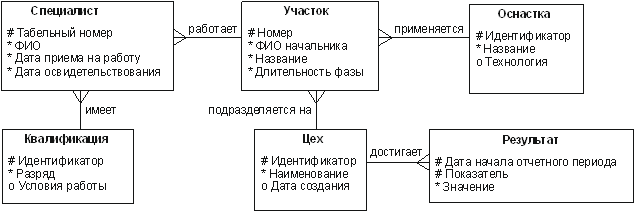
Составить SQL-предложения для создания таблиц по заданной диаграмме.

23. Дано:



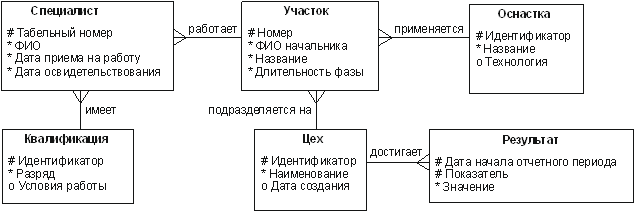
Разработать SQL-предложения создания таблиц по диаграмме.

24. Дано:



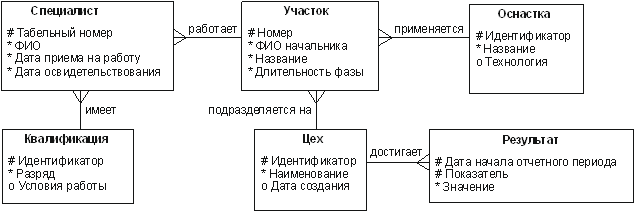
Разработать SQL-предложения для связи таблиц ЦЕХ с УЧАСТОК и УЧАСТОК со СПЕЦИАЛИСТ.

25. Дано:



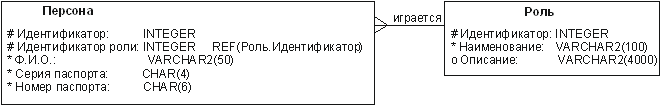
Разработать SQL-предложения для связи таблиц ОСНАСТКА с УЧАСТОК и УЧАСТОК со СПЕЦИАЛИСТ.

26. Дано:



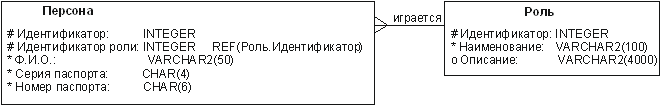
Разработать SQL-предложения для связи таблиц КВАЛИФИКАЦИЯ со СПЕЦИАЛИСТ и УЧАСТОК со СПЕЦИАЛИСТ.

27. Дано:



Разработать SQL-предложения для вставки в базу данных актеров, сыгравших роль Джеймса Бонда – главную роль серии кинофильмов с общим названием «Бондиана». Для справки: Джордж Лэзенби (1Н35X 09321008), Шон Коннери (23J11 S4500239), Тимоти Далтон (3АА3F 93001290DA), Дэниел Крейг (14FF5 3SS56120). Остальных не учитывать.

28. Дано:



Разработать SQL-предложения для вставки в базу данных актеров, сыгравших роли Дэниела Оушена (Фрэнк Синатра, Джордж Клуни) и его жену Тэсс Оушен (Энджи Дикинсон, Джулия Робертс) в фильме и его ремейке «11 друзей Оушена». Серии и номера паспортов задать любые.

29. Дано:

**ВАЛЮТА**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Идентификатор | Международный код | Наименование | Страна-эмитент | Дата начала обращения | Конвертируемость |
| 10134 | AUD | Австралийский доллар | Австралийский союз | 1963 | true |
| 3570 | BSD | Багамский доллар | Багамы | 1966 | false |
| 20034 | BBD | Барбадосский доллар | Барбадос | 1973 | true |
| 2190 | GYD | Гайанский доллар | Гайана | 1965 | false |
| 4060 | CAD | Канадский доллар | Канада | 1858 | true |
| 41003 | NZD | Новозеландский доллар | Новая Зеландия | 1967 | false |
| 15 | USD | Доллар США | США | 1792 | false |

Составить SQL-предложения для изменения значения конвертируемости валюты на противоположное значение для Новозеландского доллара, Доллара США и Барбадосского доллара, а также для изменения года начала обращения валют, датированных 1966 и 1967 годом, на 1968 год.

30. Дано:

**ВАЛЮТА**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Идентификатор | Международный код | Наименование | Страна-эмитент | Дата начала обращения | Конвертируемость |
| 10134 | AUD | Австралийский доллар | Австралийский союз | 1963 | true |
| 3570 | BSD | Багамский доллар | Багамы | 1966 | false |
| 20034 | BBD | Барбадосский доллар | Барбадос | 1973 | true |
| 2190 | GYD | Гайанский доллар | Гайана | 1965 | false |
| 4060 | CAD | Канадский доллар | Канада | 1858 | true |
| 41003 | NZD | Новозеландский доллар | Новая Зеландия | 1967 | false |
| 15 | USD | Доллар США | США | 1792 | false |

Составить SQL-предложения для изменения значения страны-эмитента на Австралия и даты начала обращения на 1971 год для Австралийского доллара, изменения международного кода на GID и наименования на «Доллар свободной Гайаны» для валюты с датой начала обращения в 1965 году, удаления значения конвертируемости для всех не конвертируемых валют.

31. Дано:

**ВАЛЮТА**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Идентификатор | Международный код | Наименование | Страна-эмитент | Дата начала обращения | Конвертируемость |
| 10134 | AUD | Австралийский доллар | Австралийский союз | 1963 | true |
| 3570 | BSD | Багамский доллар | Багамы | 1966 | false |
| 20034 | BBD | Барбадосский доллар | Барбадос | 1973 | true |
| 2190 | GYD | Гайанский доллар | Гайана | 1965 | false |
| 4060 | CAD | Канадский доллар | Канада | 1858 | true |
| 41003 | NZD | Новозеландский доллар | Новая Зеландия | 1967 | false |
| 15 | USD | Доллар США | США | 1792 | false |

Составить SQL-предложения для удаления из реестра всех валют, которые имеют значение идентификатора меньшее, чем 4000. Сколько валют будет удалено?

32. Дано:

**ВАЛЮТА**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Идентификатор | Международный код | Наименование | Страна-эмитент | Дата начала обращения | Конвертируемость |
| 10134 | AUD | Австралийский доллар | Австралийский союз | 1963 | true |
| 3570 | BSD | Багамский доллар | Багамы | 1966 | false |
| 20034 | BBD | Барбадосский доллар | Барбадос | 1973 | true |
| 2190 | GYD | Гайанский доллар | Гайана | 1965 | false |
| 4060 | CAD | Канадский доллар | Канада | 1858 | true |
| 41003 | NZD | Новозеландский доллар | Новая Зеландия | 1967 | false |
| 15 | USD | Доллар США | США | 1792 | false |

Составить SQL-предложения для удаления из реестра всех валют, которые имеют значение идентификатора меньшее, чем 4000 и являются конвертируемыми. Сколько валют будет удалено?

33. Дано:

**ВАЛЮТА**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Идентификатор | Международный код | Наименование | Страна-эмитент | Дата начала обращения | Конвертируемость |
| 10134 | AUD | Австралийский доллар | Австралийский союз | 1963 | true |
| 3570 | BSD | Багамский доллар | Багамы | 1966 | false |
| 20034 | BBD | Барбадосский доллар | Барбадос | 1973 | true |
| 2190 | GYD | Гайанский доллар | Гайана | 1965 | false |
| 4060 | CAD | Канадский доллар | Канада | 1858 | true |
| 41003 | NZD | Новозеландский доллар | Новая Зеландия | 1967 | false |
| 15 | USD | Доллар США | США | 1792 | false |

Составить SQL-предложения для вставки в реестр валюты Российской Федерации Российский рубль с идентификатором 14984, датой начала обращения с 1991 года и международным кодом RUR. Конвертируемость не указана. SQL-предложение для вставки в реестр валюты Беларуси без указания ее наименования с международным кодом BLR, датой начала обращения 1991 год, идентификатором 14991 и значением конвертируемости false.

34. Дано:

**ВАЛЮТА**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Идентификатор | Международный код | Наименование | Страна-эмитент | Дата начала обращения | Конвертируемость |
| 10134 | AUD | Австралийский доллар | Австралийский союз | 1963 | true |
| 3570 | BSD | Багамский доллар | Багамы | 1966 | false |
| 20034 | BBD | Барбадосский доллар | Барбадос | 1973 | true |
| 2190 | GYD | Гайанский доллар | Гайана | 1965 | false |
| 4060 | CAD | Канадский доллар | Канада | 1858 | true |
| 41003 | NZD | Новозеландский доллар | Новая Зеландия | 1967 | false |
| 15 | USD | Доллар США | США | 1792 | false |

Составить SQL-предложения для вставки в реестр валюты Республики Польша (Польский злотый) PLN с идентификатором 985, датой начала обращения с 1924 года и возможностью ее конвертации. SQL-предложение для вставки в реестр валюты Великобритании (Фунт стерлингов) с международным кодом GBP, датой начала обращения 1694 год, идентификатором 826, без указания значения конвертируемости.

35. Дано:

**ВАЛЮТА**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Идентификатор | Международный код | Наименование | Страна-эмитент | Дата начала обращения | Конвертируемость |
| 10134 | AUD | Австралийский доллар | Австралийский союз | 1963 | true |
| 3570 | BSD | Багамский доллар | Багамы | 1966 | false |
| 20034 | BBD | Барбадосский доллар | Барбадос | 1973 | true |
| 2190 | GYD | Гайанский доллар | Гайана | 1965 | false |
| 4060 | CAD | Канадский доллар | Канада | 1858 | true |
| 41003 | NZD | Новозеландский доллар | Новая Зеландия | 1967 | false |
| 15 | USD | Доллар США | США | 1792 | false |

Составить SQL-предложения для вставки в реестр валюты Швейцарской конфедерации CHF с идентификатором 4060, датой начала обращения с 1850 года и возможностью ее конвертации (наименование валюты не указывать). SQL-предложение для изменения в реестре международного кода валюты с идентификатором 4060 на GBP, даты начала ее обращения на 1694 год. Разработать SQL-предложение по удалению из реестра всех конвертируемых валют.

Для лентяев

Скрипт создания и заполнения учебной базы данных по домашним заданиям на семинары 13 и 14

**Для СУБД ORACLE (APEX):**

**CREATE TABLE Marka (IDM SMALLINT NOT NULL, Producer VARCHAR2(15) NOT NULL, Model VARCHAR2(15) NOT NULL, Country CHAR(2));**

**CREATE TABLE Autos (Gosn VARCHAR2(9) NOT NULL, IDM SMALLINT NOT NULL, DateB DATE, Distance INTEGER, Category VARCHAR2(50), Price INTEGER);**

**ALTER TABLE Marka ADD PRIMARY KEY (IDM);**

**ALTER TABLE Autos ADD PRIMARY KEY (IDM, Gosn);**

**ALTER TABLE Autos ADD FOREIGN KEY (IDM) REFERENCES Marka(IDM);**

**ALTER TABLE Autos RENAME TO Vehicle;**

**INSERT INTO Marka(IDM, Producer, Model, Country) VALUES(1, 'Mercedes-Benz', '600SEL', 'DE');**

**INSERT INTO Marka(IDM, Producer, Model, Country) VALUES(2, 'Audi', 'R8', 'DE');**

**INSERT INTO Marka(IDM, Producer, Model, Country) VALUES(3, 'Mercedes-Benz', 'CLA200', 'DE');**

**INSERT INTO Marka(IDM, Producer, Model, Country) VALUES(4, 'Volkswagen', 'Passat', 'DE');**

**INSERT INTO Marka(IDM, Producer, Model, Country) VALUES(5, 'Ford', 'Focus', 'US');**

**INSERT INTO Marka(IDM, Producer, Model, Country) VALUES(6, 'Toyota', 'Corolla', 'JP');**

**INSERT INTO Marka(IDM, Producer, Model, Country) VALUES(7, 'Lexus', 'NX240', 'JP');**

**INSERT INTO Marka(IDM, Producer, Model, Country) VALUES(8, 'KIA', 'K5', 'KR');**

**SELECT \* FROM Marka;**

**INSERT INTO Vehicle(IDM, Gosn, DateB, Distance, Category, Price)**

**VALUES(1, 'y309ak777', to\_date('25.03.2021', 'dd.mm.yyyy'), 28934, 'VIP-Такси', 3450000);**

**INSERT INTO Vehicle(IDM, Gosn, DateB, Distance, Category, Price)**

**VALUES(1, 'a355co797', to\_date('20.05.2020', 'dd.mm.yyyy'), 86004, 'VIP-Такси', 3650000);**

**INSERT INTO Vehicle(IDM, Gosn, DateB, Distance, Category, Price)**

**VALUES(4, 'c006ka799', to\_date('01.10.2021', 'dd.mm.yyyy'), 39644, 'VIP-Такси', 1490000);**

**INSERT INTO Vehicle(IDM, Gosn, DateB, Distance, Category, Price)**

**VALUES(4, 'o097cc77', to\_date('15.08.2019', 'dd.mm.yyyy'), 92034, 'Сопровождение', 1560000);**

**INSERT INTO Vehicle(IDM, Gosn, DateB, Distance, Category, Price)**

**VALUES(2, 'o991ac797', to\_date('23.11.2020', 'dd.mm.yyyy'), 72301, 'Сопровождение', 4500000);**

**INSERT INTO Vehicle(IDM, Gosn, DateB, Distance, Category, Price)**

**VALUES(2, 'a580ac777', to\_date('15.12.2020', 'dd.mm.yyyy'), 44962, 'Сопровождение', 3990000);**

**INSERT INTO Vehicle(IDM, Gosn, DateB, Distance, Category, Price)**

**VALUES(3, 'x544ak797', to\_date('11.04.2020', 'dd.mm.yyyy'), 98701, 'Сопровождение', 3550000);**

**INSERT INTO Vehicle(IDM, Gosn, DateB, Distance, Category, Price)**

**VALUES(3, 'c092ak177', to\_date('26.01.2018', 'dd.mm.yyyy'), 129005, 'Эскорт', 2590000);**

**INSERT INTO Vehicle(IDM, Gosn, DateB, Distance, Category, Price)**

**VALUES(3, 'y915kx177', to\_date('17.03.2021', 'dd.mm.yyyy'), 46980, 'Эскорт', 3600000);**

**INSERT INTO Vehicle(IDM, Gosn, DateB, Distance, Category, Price)**

**VALUES(5, 'y631kb790', to\_date('22.02.2018', 'dd.mm.yyyy'), 59945, 'Эскорт', 1050000);**

**INSERT INTO Vehicle(IDM, Gosn, DateB, Distance, Category, Price)**

**VALUES(5, 'y659kb790', to\_date('24.05.2018', 'dd.mm.yyyy'), 58701, 'Поручения', 1450000);**

**INSERT INTO Vehicle(IDM, Gosn, DateB, Distance, Category, Price)**

**VALUES(6, 'e803ok177', to\_date('05.12.2021', 'dd.mm.yyyy'), 9934, 'Поручения', 1500000);**

**INSERT INTO Vehicle(IDM, Gosn, DateB, Distance, Category, Price)**

**VALUES(6, 'k135yk799', to\_date('12.06.2020', 'dd.mm.yyyy'), 34603, 'Поручения', 1450000);**

**INSERT INTO Vehicle(IDM, Gosn, DateB, Distance, Category, Price)**

**VALUES(7, 'c290ax797', to\_date('22.09.2020', 'dd.mm.yyyy'), 89314, 'Поручения', 3150000);**

**INSERT INTO Vehicle(IDM, Gosn, DateB, Distance, Category, Price)**

**VALUES(7, 'e655ak199', to\_date('06.08.2020', 'dd.mm.yyyy'), 99802, 'Поручения', 3550000);**

**INSERT INTO Vehicle(IDM, Gosn, DateB, Distance, Category, Price)**

**VALUES(8, 'c501ac799', to\_date('20.12.2020', 'dd.mm.yyyy'), 62304, 'Поручения', 2050000);**

**SELECT \* FROM Vehicle;**

**Для СУБД PostgreSQL:**

**CREATE TABLE Marka (IDM SMALLINT NOT NULL, Producer VARCHAR(15) NOT NULL, Model VARCHAR(15) NOT NULL, Country CHAR(2));**

**CREATE TABLE Autos (Gosn VARCHAR(9) NOT NULL, IDM SMALLINT NOT NULL, DateB DATE, Distance INTEGER, Category VARCHAR2(50), Price INTEGER);**

**ALTER TABLE Marka ADD PRIMARY KEY (IDM);**

**ALTER TABLE Autos ADD PRIMARY KEY (IDM, Gosn);**

**ALTER TABLE Autos ADD FOREIGN KEY (IDM) REFERENCES Marka(IDM);**

**ALTER TABLE Autos RENAME TO Vehicle;**

**INSERT INTO Marka(IDM, Producer, Model, Country) VALUES(1, 'Mercedes-Benz', '600SEL', 'DE');**

**INSERT INTO Marka(IDM, Producer, Model, Country) VALUES(2, 'Audi', 'R8', 'DE');**

**INSERT INTO Marka(IDM, Producer, Model, Country) VALUES(3, 'Mercedes-Benz', 'CLA200', 'DE');**

**INSERT INTO Marka(IDM, Producer, Model, Country) VALUES(4, 'Volkswagen', 'Passat', 'DE');**

**INSERT INTO Marka(IDM, Producer, Model, Country) VALUES(5, 'Ford', 'Focus', 'US');**

**INSERT INTO Marka(IDM, Producer, Model, Country) VALUES(6, 'Toyota', 'Corolla', 'JP');**

**INSERT INTO Marka(IDM, Producer, Model, Country) VALUES(7, 'Lexus', 'NX240', 'JP');**

**INSERT INTO Marka(IDM, Producer, Model, Country) VALUES(8, 'KIA', 'K5', 'KR');**

**SELECT \* FROM Marka;**

**INSERT INTO Vehicle(IDM, Gosn, DateB, Distance, Category, Price)**

**VALUES(1, 'y309ak777', to\_date('25.03.2021', 'dd.mm.yyyy'), 28934, 'VIP-Такси', 3450000);**

**INSERT INTO Vehicle(IDM, Gosn, DateB, Distance, Category, Price)**

**VALUES(1, 'a355co797', to\_date('20.05.2020', 'dd.mm.yyyy'), 86004, 'VIP-Такси', 3650000);**

**INSERT INTO Vehicle(IDM, Gosn, DateB, Distance, Category, Price)**

**VALUES(4, 'c006ka799', to\_date('01.10.2021', 'dd.mm.yyyy'), 39644, 'VIP-Такси', 1490000);**

**INSERT INTO Vehicle(IDM, Gosn, DateB, Distance, Category, Price)**

**VALUES(4, 'o097cc77', to\_date('15.08.2019', 'dd.mm.yyyy'), 92034, 'Сопровождение', 1560000);**

**INSERT INTO Vehicle(IDM, Gosn, DateB, Distance, Category, Price)**

**VALUES(2, 'o991ac797', to\_date('23.11.2020', 'dd.mm.yyyy'), 72301, 'Сопровождение', 4500000);**

**INSERT INTO Vehicle(IDM, Gosn, DateB, Distance, Category, Price)**

**VALUES(2, 'a580ac777', to\_date('15.12.2020', 'dd.mm.yyyy'), 44962, 'Сопровождение', 3990000);**

**INSERT INTO Vehicle(IDM, Gosn, DateB, Distance, Category, Price)**

**VALUES(3, 'x544ak797', to\_date('11.04.2020', 'dd.mm.yyyy'), 98701, 'Сопровождение', 3550000);**

**INSERT INTO Vehicle(IDM, Gosn, DateB, Distance, Category, Price)**

**VALUES(3, 'c092ak177', to\_date('26.01.2018', 'dd.mm.yyyy'), 129005, 'Эскорт', 2590000);**

**INSERT INTO Vehicle(IDM, Gosn, DateB, Distance, Category, Price)**

**VALUES(3, 'y915kx177', to\_date('17.03.2021', 'dd.mm.yyyy'), 46980, 'Эскорт', 3600000);**

**INSERT INTO Vehicle(IDM, Gosn, DateB, Distance, Category, Price)**

**VALUES(5, 'y631kb790', to\_date('22.02.2018', 'dd.mm.yyyy'), 59945, 'Эскорт', 1050000);**

**INSERT INTO Vehicle(IDM, Gosn, DateB, Distance, Category, Price)**

**VALUES(5, 'y659kb790', to\_date('24.05.2018', 'dd.mm.yyyy'), 58701, 'Поручения', 1450000);**

**INSERT INTO Vehicle(IDM, Gosn, DateB, Distance, Category, Price)**

**VALUES(6, 'e803ok177', to\_date('05.12.2021', 'dd.mm.yyyy'), 9934, 'Поручения', 1500000);**

**INSERT INTO Vehicle(IDM, Gosn, DateB, Distance, Category, Price)**

**VALUES(6, 'k135yk799', to\_date('12.06.2020', 'dd.mm.yyyy'), 34603, 'Поручения', 1450000);**

**INSERT INTO Vehicle(IDM, Gosn, DateB, Distance, Category, Price)**

**VALUES(7, 'c290ax797', to\_date('22.09.2020', 'dd.mm.yyyy'), 89314, 'Поручения', 3150000);**

**INSERT INTO Vehicle(IDM, Gosn, DateB, Distance, Category, Price)**

**VALUES(7, 'e655ak199', to\_date('06.08.2020', 'dd.mm.yyyy'), 99802, 'Поручения', 3550000);**

**INSERT INTO Vehicle(IDM, Gosn, DateB, Distance, Category, Price)**

**VALUES(8, 'c501ac799', to\_date('20.12.2020', 'dd.mm.yyyy'), 62304, 'Поручения', 2050000);**

**SELECT \* FROM Vehicle;**