**Вопрос №5**

**1. Язык SQL. Предложение SELECT оператора SELECT.**

**SELECT** — оператор запроса в языке SQL, возвращающий набор данных (выборку) из базы данных.

**SELECT (Предложение)** – определяет список возвращаемых столбцов (как существующих, так и вычисляемых).

**FROM** - задаёт табличное выражение, которое определяет базовый набор данных для применения операций, определяемых в других приложениях оператора **SELECT**;



**2. Язык SQL. Предложение WHERE оператора SELECT.**

**WHERE** - задает ограничение на строки табличного выражения из предложения **FROM**

*Пример.*

Выбрать студентов, получающих стипендию, равную 150.

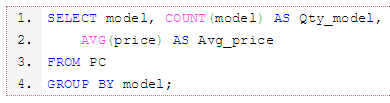
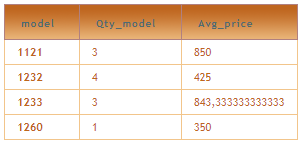
**SELECT fname FROM stident**

**WHERE stip=150;**

С помощью данного SQL запроса SELECT выбираются все значения из таблицы STUDENT, поле STIP которых строго равно 150.

**3. Язык SQL. Предложение GROUP BY оператора SELECT.**

**GROUP BY** - используется для определения групп выходных строк, к которым могут применяться агрегатные функции (**COUNT**, **MIN**, **MAX**, **AVG** и **SUM**)

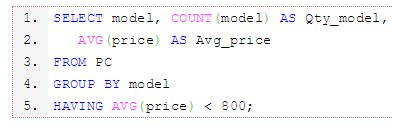
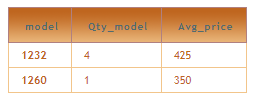
В этом запросе для каждой модели ПК определяется их количество и средняя стоимость. Все строки с одинаковыми значениями model (номер модели) образуют группу, и на выходе **SELECT**вычисляются количество значений и средняя цена для каждой группы.

**4. Язык SQL. Предложение HAVING оператора SELECT.**

**HAVING** - аналогичен **WHERE** за исключением того, что строки отбираются не по значениям столбцов, а строятся из значений столбцов, указанных в **GROUP BY**, и значений агрегатных функций, вычисленных для каждой группы, образованной **GROUP BY**.

*Пример.*

Получить количество ПК и среднюю цену для каждой модели, средняя цена которой менее $800.

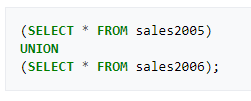
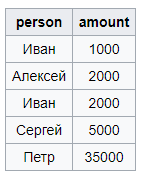
**5. Язык SQL. Конструкция UNION оператора SELECT.**

**UNION** - объединяет результаты двух или более запросов в один результирующий набор, в который входят все строки, принадлежащие всем запросам в объединении.

Даны две таблицы:

При выполнении следующего запроса, получаем следующий набор данных:

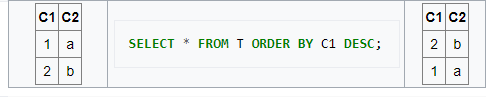
 

**6. Язык SQL. Предложение ORDER BY оператора SELECT.**

**ORDER BY** - задает критерии сортировки результирующего набора строк.

**ASC (по умолчанию)** - устанавливает порядок сортирования по возрастанию, от меньших значений к большим.

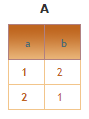
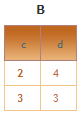
**DESC** - устанавливает порядок сортирования по убыванию, от больших значений к меньшим.



**7. Язык SQL. Предложение FROM оператора SELECT. Декартово произведение и внутреннее соединение.**

В предложении **FROM** допускается указание нескольких таблиц. Простое перечисление таблиц через запятую практически не используется, поскольку оно соответствует реляционной операции, которая называется декартовым произведением.

*Пример.*

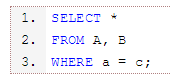
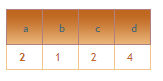
 

Результат запроса будет выглядеть следующим образом:

Поэтому перечисление таблиц, как правило, используется совместно с условием соединения строк из разных таблиц, указываемым в предложении WHERE.

*Теперь результатом выполнения этого запроса будет следующая таблица:*

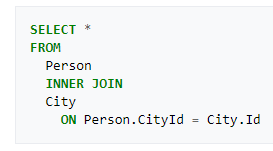
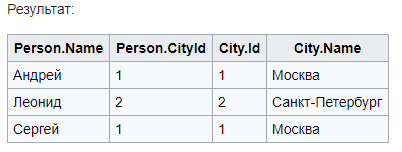
 

**INNER JOIN** – оператор внутреннего соединения, соединяющий две таблицы. Порядок таблиц для оператора неважен, поскольку оператор является симметричным.

*Тело результата логически формируется следующим образом.*

Каждая строка одной таблицы сопоставляется с каждой строкой второй таблицы, после чего для полученной «соединённой» строки проверяется условие соединения (вычисляется предикат соединения). Если условие истинно, в таблицу-результат добавляется соответствующая «соединённая» строка.

**8. Язык SQL. Предложение FROM оператора SELECT. Левое внешнее и правое внешнее соединение.**

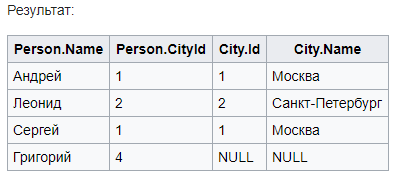
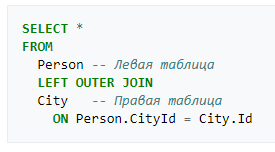


**LEFT OUTER JOIN** – оператор левого внешнего соединения, соединяющий две таблицы. Порядок таблиц для оператора важен, поскольку оператор не является симметричным.

*Тело результата логически формируется следующим образом.* Пусть выполняется соединение левой и правой таблиц по предикату (условию) p.

1) В результат включается внутреннее соединение (INNER JOIN) левой и правой таблиц по предикату p.

2) Затем в результат добавляются те записи левой таблицы, которые не вошли во внутреннее соединение на шаге 1. Для таких записей поля, соответствующие правой таблице, заполняются значениями NULL.

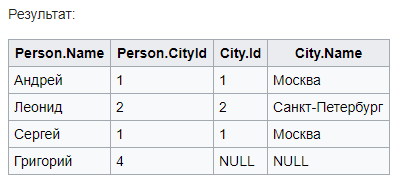
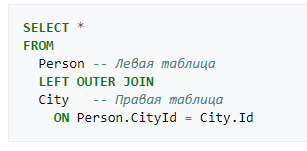


**RIGHT OUTER JOIN** – оператор правого внешнего соединения, соединяющий две таблицы. Порядок таблиц для оператора важен, поскольку оператор не является симметричным.

*Тело результата логически формируется следующим образом.* Пусть выполняется соединение левой и правой таблиц по предикату (условию) p.

1) В результат включается внутреннее соединение (INNER JOIN) левой и правой таблиц по предикату p.

2) Затем в результат добавляются те записи правой таблицы, которые не вошли во внутреннее соединение на шаге 1. Для таких записей поля, соответствующие левой таблице, заполняются значениями NULL.



**9. Язык SQL. Использование подзапросов.**

*Подзапрос* — это запрос, содержащийся в выражении ключевого слова **WHERE** другого запроса с целью дополнительных ограничений на выводимые данные.

**SELECT** имя\_столбиа **FROM** таблица

**WHERE** имя\_столбца =

(**SELECT** имя\_столбца **FROM** таблица

**WHERE** условия);\

**10. Язык SQL. Оператор INSERT.**

**INSERT** — оператор языка SQL, который позволяет добавить строки в таблицу, заполняя их значениями.

Использование перечисления значений с указанием столбцов:

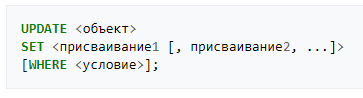


Использование перечисления значений без указания столбцов:



**11. Язык SQL. Оператор UPDATE.**

**UPDATE** — оператор языка SQL, позволяющий обновить значения в заданных столбцах таблицы.



* <объект> — объект, над которым выполняется действие (таблица или [представление](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%B5%D0%B4%D1%81%D1%82%D0%B0%D0%B2%D0%BB%D0%B5%D0%BD%D0%B8%D0%B5_(%D0%B1%D0%B0%D0%B7%D1%8B_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85)))
* <присваивание> — присваивание, которое будет выполняться при каждом выполнении условия <условие>, или для каждой записи, если отсутствует раздел where
* <условие> — условие выполнения команды
* SET — после ключевого слова должен идти список полей таблицы, которые будут обновлены и непосредственно сами новые значения в виде

имя поля="значение"

**12. Язык SQL. Оператор DELETE.**

**DELETE** — операция удаления записей из таблицы.

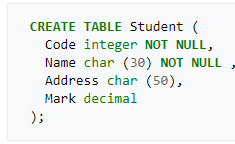
*Общий синтаксис команды:*



**13. Язык SQL. Создание, удаление и изменение структуры таблиц.**

**CREATE TABLE** — оператор языка SQL, используемый для создания таблиц базы данных.

*Пример.*



**ALTER TABLE** - изменяет определение таблицы путем изменения, добавления или удаления столбцов.

**DROP TABLE** - удаляет одно или больше определений таблиц и все данные, индексы, триггеры, ограничения и разрешения для этих таблиц.

**14. Язык SQL. Создание ограничения первичного ключа.**

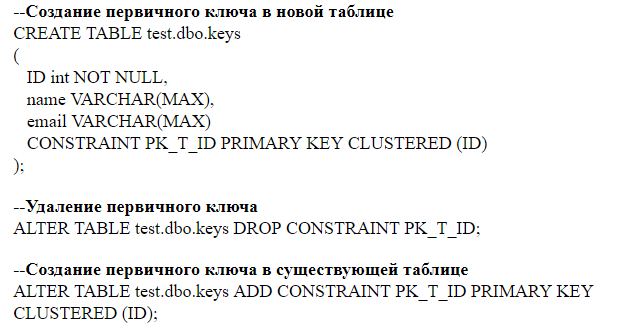
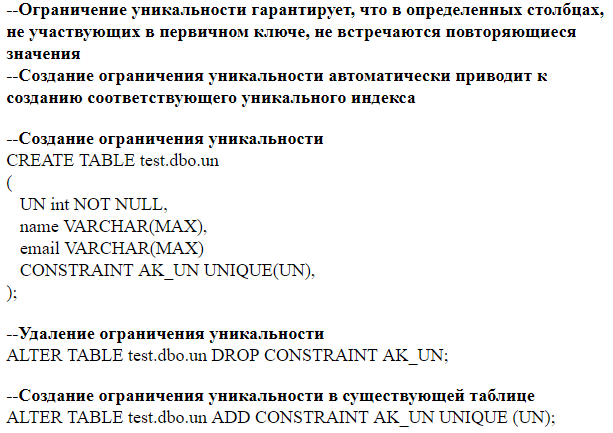
**PRIMARY KEY** - есть столбец или комбинация столбцов, содержащих значения, уникально определяющие каждую строку таблицы.

*Ограничения*

1) В таблице возможно наличие только одного ограничения по первичному ключу.

2) Все столбцы с ограничением **PRIMARY KEY** должны иметь признак **NOT NULL**. Если допустимость значения **NULL** не указана, то для всех столбцов c ограничением **PRIMARY KEY** устанавливается признак **NOT** **NULL**.

3) Первичный ключ не может включать больше 16 столбцов.

**15. Язык SQL. Создание ограничения внешнего ключа.**

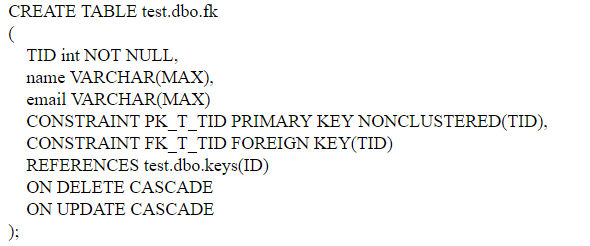
*Внешний ключ (FK)* — это столбец или сочетание столбцов, которое применяется для принудительного установления связи между данными в двух таблицах.

**Создание связей внешнего ключа. Связь создается между двумя таблицами, чтобы связать строки одной таблицы со строками другой.**

***Создание внешнего ключа в новой таблице***

**Предложения ON DELETE CASCADE и ON UPDATE CASCADE обеспечевают распространение изменений, вносимых в таблицу keys на таблицу fk**

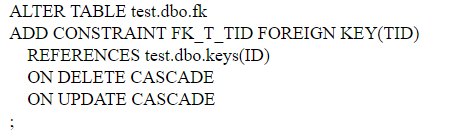
**ON DELETE CASCADE - при удалении строк и ON UPDATE CASCADE при изменении соответственно**



**Удаление внешнего ключа**



**Создание внешнего ключа в существующей таблице  
 ADD CONSTRAINT таблица\_с\_внешним\_ключом FOREIGN KEY(столбец) REFERENCES основная\_таблица (столбец)**



**16. Язык SQL. Создание ограничений умолчания, проверки и уникальности.**

**NOT NULL** – ограничение, предохраняющее поле от разрешения в нем пустых (NULL) значений

**CHECK** – ограничение, указывающее значения данных, допустимые в одном или нескольких столбцах.

**UNIQUE** – ограничение, которое можно использовать, чтобы убедиться, что в отдельные столбцы, не включенные в первичный ключ, не вводятся повторяющиеся значения.

**17. Метод ER-диаграмм: основные понятия.**

*Основными понятиями метода сущность-связь являются следующие:*

•  сущность,

•  атрибут сущности,

•  ключ сущности,

•  связь между сущностями,

•  степень связи,

•  класс принадлежности экземпляров сущности,

•  диаграммы ER-экземпляров,

•  диаграммы ER-типа.

*Сущность*представляет собой объект, информация о котором хранится в БД.

*Атрибут*представляет собой свойство сущности. Это понятие аналогично понятию атрибута в отношении.

*Ключ сущности -*атрибут или набор атрибутов, используемый для иден­тификации экземпляра сущности.

*Связь двух или более сущностей -*предполагает зависимость между атрибута­ми этих сущностей.

С целью повышения наглядности и удобства проектирования для пред­ставления сущностей, экземпляров сущностей и связей между ними исполь­зуются следующие графические средства:

•  *диаграммы ER-экземпляров,*

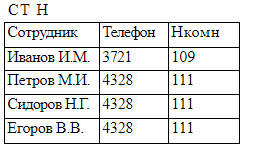
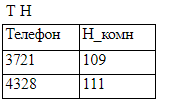
•  *диаграммы ER-muna,*или *ER-диаграммы.*

*Степень связи*является характеристикой связи между сущностями, кото­рая может быть типа: 1:1, 1:М, М:1, М:М.

**18. Избыточное дублирование и аномалии в отношениях БД.**

*Избыточное дублирование*

*Пример. Решение проблемы:*

***Аномалии***

1) Модификации 2) Удаления 3) Добавления

*Аномалии модификации*проявляются в том, что изменение значения одно­го данного может повлечь за собой просмотр всей таблицы и соответствую­щее изменение некоторых других записей таблицы.

Так, например, изменение номера телефона в комнате 111 (рис. 5.2а), что представляет собой один единственный факт, потребует просмотра всей таб­лицы С\_Т\_Н и изменения поля Нкомн согласно текущему содержимому таблицы в записях, относящихся к Петрову, Сидорову и Егорову.

*Аномалии удаления*состоят в том, что при удалении какого-либо данного из таблицы может пропасть и другая информация, которая не связана напрямую с удаляемым данным.

В той же таблице С\_Т\_Н удаление записи о сотруднике Иванове (напри­мер, по причине увольнения или ухода на заслуженный отдых) приводит к ис­чезновению информации о номере телефона, установленного в 109-й комнате.

*Аномалии добавления*возникают в случаях, когда информацию в таблицу нельзя поместить до тех пор, пока она неполная, либо вставка новой записи требует дополнительного просмотра таблицы.

**19. Зависимость между атрибутами в отношениях БД.**

*Функциональная взаимозависимость.*

Если существует функциональная зависимость вида А->В и В->А, то между А и В имеется взаимно однозначное соответствие, или функциональная взаимозависимость.

*Частичной зависимостью* - называется зависимость неключевого атрибута от части составного ключа.

*Полная функциональная зависимость* – это зависимость неключевого атрибута от всего составного ключа.

Атрибут С зависит от атрибута А *транзитивно* (существует *транзитивная зависимость*), если для атрибутов А, В, С выполняются условия А->В и В->С, но обратная зависимость отсутствует. В отношении Преподаватель транзитивной зависимостью связаны атрибуты:

ФИО->Должн->Оклад

**20. Метод нормальных форм. 1НФ.**

**Первая нормальная форма** требует, чтобы на любом пересечении строки и столбца находилось единственное значение, которое должно быть атомарным (неделимым). В таблице, удовлетворяющей 1НФ, не должно быть повторяющихся групп.

**21. Метод нормальных форм. 2НФ.**

**Вторая нормальная форма** основана на понятии полной функциональной зависимости. Оно требует, чтобы любой неключевой столбец зависел от всего первичного ключа, а не от его отдельных компонентов. Это правило относится к случаю, когда первичный ключ образован из нескольких столбцов. Первичные ключи всех таблиц из учебной базы данных являются простыми (состоят из одного столбца), поэтому все таблицы находятся не только в 1НФ, но и однозначно во 2НФ.

**22. Метод нормальных форм. 3НФ.**

**Третья нормальная форма** основана на понятии транзитивной *зависимости*. Если для атрибутов А, В и С некоторого отношения существуют зависимости С от В и В от А, то говорят, что атрибут С транзитивно зависит от атрибута А через атрибут В.

Отношение находится в 3НФ, если оно находится в 1НФ и 2НФ, и в нем не существует транзитивных зависимостей неключевых атрибутов от первичного ключа. *Другими словами, третья нормальная форма требует, чтобы ни один неключевой столбец не зависел бы от другого неключевого столбца. Любой неключевой столбец должен зависеть только от столбца первичного ключа.*

**23. Метод нормальных форм. Нормальная форма Бойса-Кодда.**

**Нормальная форма Бойса-Кодда** учитывает функциональные зависимости, в которых участвуют все потенциальные ключи отношения, а не только его первичный ключ. Для отношения с единственным потенциальным ключом 3НФ и НФБК эквивалентны.

Отношение находится в НФБК тогда и только тогда, когда каждый его

детерминант является потенциальным ключом.

*Детерминант* – это атрибут или группа атрибутов от которых зависят другие атрибуты.

**24. Особенности проектирования БД для OLTP и OLAP**

*OLAP* - это Online Analytical Processing, т. е. оперативный анализ данных.

*Термин определяет категорию приложений и технологий, которые обеспечивают сбор, хранение, манипулирование и анализ многомерных данных.*

OLTP (англ. Online Transaction Processing), транзакционная система — обработка транзакций в реальном времени.

Способ организации БД, при котором система работает с небольшими по размерам транзакциями, но идущими большим потоком, и при этом клиенту требуется от системы минимальное время отклика.

**25. Понятие транзакции.**

*Транзакцией* - это последовательность операций над БД, выполняемая как единое целое. Транзакция переводит БД из одного непротиворечивого состояния в другое непротиворечивое состояние.