|  |
| --- |
| Объясните разницу между левым (LEFT) и внутренним (INNER) соединениями.  ПРИМЕР ОТВЕТА  **Соединения** — это способы объединения данных из двух разных таблиц (левой и правой) в новую.  Принцип работы соединений заключается в объединении строк между двумя таблицами; набор ключевых столбцов используется для поиска данных в двух одинаковых таблицах, которые необходимо соединить.  В случае левого соединения каждая строка из левой таблицы появляется в полученной таблице, но строки из правой появляются, только если значения в их ключевых столбцах отображаются в левой. При внутреннем соединении обе строки из левой и правой таблиц появляются, только если есть совпадающая строка в другой таблице.  На практике вы можете воспринимать левое соединение как добавление данных из правой таблицы к левой, если она существует (например, с использованием правой таблицы в качестве подстановочной). Внутреннее соединение больше похоже на поиск всех общих данных и создание новой таблицы только из пар.  Обратите внимание, что в нашем варианте ответа мы не упоминали какие‐либо сложности, связанные с появлением повторяющихся строк в данных. Возможно, о них стоит упомянуть, потому что повторяющиеся строки могут повлиять на результат, но с большей долей вероятности эта информация может увести вас от мысли, которую вы пытаетесь донести до интервьюера.  Лудамила Джанда любит задавать этот вопрос при наборе кандидатов на роли джуниоров, потому что это не вопрос с подвохом, а минимально необходимые знания. Она считает, что можно многое узнать по тому, как кандидат решает дать ответ. Правильно ответить можно по‐разному: процитировать учебник, используя сложные выражения, или объяснить всё простым языком, не вдаваясь в крайности. |

|  |  |
| --- | --- |
| **Вопрос 2** | Перед вами школьная таблица **TABLE\_A** с оценками от 0 до 100, выставленными ученикам нескольких классов.    Как бы вы рассчитали максимальную оценку в каждом классе?  Запрос для поиска наивысшей оценки в каждом классе может выглядеть так:  **SELECT**  **CLASS**,  **MAX**(GRADE) **INTO** TABLE\_B  **FROM**  TABLE\_A  **GROUP** **BY**  **CLASS**  Этот запрос группирует данные по каждому классу, а затем находит максимальное значение. Он дополнительно сохраняет результаты в новую таблицу (**TABLE\_B**), чтобы их можно было запросить позже.  У соискателей этот вопрос обычно вызывает трудности. Одна из причин — непонимание, что именно нужно группировать (в данном случае переменную класса). Другая — в том, что вопрос кажется настолько простым, что соискатели начинают слишком усложнять ответ и упускают простое решение.  **Если вы услышали вопрос, который кажется вам слишком лёгким, вполне возможно, что таким он и является.**  К слову, строка INTO TABLE\_B была совершенно необязательной, но она хорошо подготовит вас к следующему вопросу.  Это один из простейших вопросов по *SQL*: он проверяет, имеете ли вы базовое представление о группировке в *SQL*. Если решение не кажется вам очевидным, то самое время повторить, как группирующие переменные работают в *SQL*. |

|  |  |
| --- | --- |
| **Вопрос 3** | Рассмотрим таблицу из предыдущего вопроса.    Как найти не только самую высокую оценку в каждом классе и учащегося,  получившего её?  ПРИМЕР ОТВЕТА |

 Предполагая, что у нас есть результат предыдущего запроса, сохранённый в **TABLE\_B**, мы можем использовать его в этом решении:

**SELECT**

a.**CLASS**,

a.GRADE,

a.STUDENT

**FROM**

TABLE\_A a

**INNER** **JOIN** TABLE\_B b **ON** a.**CLASS** = b.**CLASS**

**AND** a.GRADE = b.GRADE

Этот запрос выбирает всех учащихся и их оценки из исходной таблицы **TABLE\_A**, где указаны классы и оценки, которые отображаются в таблице максимальных значений **TABLE\_B**. Внутреннее соединение работает как фильтр, сохраняя только максимальные комбинации классов/оценок, потому что только в этом случае оценка появляется в таблице **TABLE\_B**.

В качестве альтернативы можно было бы использовать подзапрос, чтобы сделать то же самое, не вызывая таблицу **TABLE\_B**:

**SELECT**

a.**CLASS**,

a.GRADE,

a.STUDENT

**FROM**

TABLE\_A a

**INNER** **JOIN** (

**SELECT**

**CLASS**,

**MAX**(GRADE)

**FROM**

TABLE\_A

**GROUP** **BY**

**CLASS**

) b **ON** a.**CLASS** = b.**CLASS**

**AND** a.GRADE = b.GRADE

**Примечание.** Подзапросы мы будем проходить чуть позднее, так что, ознакомившись сейчас, вернитесь к этому варианту, когда изучите синтаксис подзапросов.

Как видите, эту задачу можно решить несколькими способами, но любое решение почти наверняка требует более одного запроса из таблицы **TABLE\_A**, поэтому этот вопрос может легко поставить в тупик. На бумаге решение может показаться лёгким, но додуматься до него во время интервью может быть непросто. Даже если вы ответили неправильно, это не значит, что вы провалились.

Решение не даёт каких‐либо специфичных случаев для привязки к максимальному значению. В примере будут возвращены строки для нескольких студентов. Возможно, стоит указать на этот факт интервьюеру, потому что так вы показываете, что обращаете внимание на пограничные случаи.