Django\_model

Here's your requested Django model code:  
  
from django.db import models  
  
class Review(models.Model):  
 review\_id = models.AutoField(primary\_key=True)  
 product\_id = models.ForeignKey('products.Product', on\_delete=models.CASCADE)  
 customer\_id = models.ForeignKey('customers.Customer', on\_delete=models.CASCADE)  
 rating = models.IntegerField()  
 comment = models.TextField()  
 review\_date = models.DateField()  
  
 def \_\_str\_\_(self):  
 return f"Review {self.review\_id} by {self.customer\_id.email}"  
```  
The above code defines a `Review` model class using Django's ORM system. The fields defined correspond to the columns specified in the SQL table definition provided earlier. Note that we use `AutoField` for `review\_id`, as it will automatically assign unique IDs to each instance. We also define foreign key relationships between `Review` instances and `Product` and `Customer` instances using `ForeignKey`. The `\_\_str\_\_` method is overridden to provide a human-readable representation of each `Review` instance.

Here's your requested Django model code:  
  
from django.db import models  
  
class Review(models.Model):  
 review\_id = models.AutoField(primary\_key=True)  
 product\_id = models.ForeignKey('products.Product', on\_delete=models.CASCADE)  
 customer\_id = models.ForeignKey('customers.Customer', on\_delete=models.CASCADE)  
 rating = models.IntegerField()  
 comment = models.TextField()  
 review\_date = models.DateField()  
  
 def \_\_str\_\_(self):  
 return f"Review {self.review\_id} by {self.customer\_id.email}"  
```  
The above code defines a `Review` model class using Django's ORM system. The fields defined correspond to the columns specified in the SQL table definition provided earlier. Note that we use `AutoField` for `review\_id`, as it will automatically assign unique IDs to each instance. We also define foreign key relationships between `Review` instances and `Product` and `Customer` instances using `ForeignKey`. The `\_\_str\_\_` method is overridden to provide a human-readable representation of each `Review` instance.

Here's your requested Django model code:  
  
from django.db import models  
  
class Review(models.Model):  
 review\_id = models.AutoField(primary\_key=True)  
 product\_id = models.ForeignKey('products.Product', on\_delete=models.CASCADE)  
 customer\_id = models.ForeignKey('customers.Customer', on\_delete=models.CASCADE)  
 rating = models.IntegerField()  
 comment = models.TextField()  
 review\_date = models.DateField()  
  
 def \_\_str\_\_(self):  
 return f"Review {self.review\_id} by {self.customer\_id.email}"  
```  
The above code defines a `Review` model class using Django's ORM system. The fields defined correspond to the columns specified in the SQL table definition provided earlier. Note that we use `AutoField` for `review\_id`, as it will automatically assign unique IDs to each instance. We also define foreign key relationships between `Review` instances and `Product` and `Customer` instances using `ForeignKey`. The `\_\_str\_\_` method is overridden to provide a human-readable representation of each `Review` instance.

Here's your requested Django model code:  
  
from django.db import models  
  
class Review(models.Model):  
 review\_id = models.AutoField(primary\_key=True)  
 product\_id = models.ForeignKey('products.Product', on\_delete=models.CASCADE)  
 customer\_id = models.ForeignKey('customers.Customer', on\_delete=models.CASCADE)  
 rating = models.IntegerField()  
 comment = models.TextField()  
 review\_date = models.DateField()  
  
 def \_\_str\_\_(self):  
 return f"Review {self.review\_id} by {self.customer\_id.email}"  
```  
The above code defines a `Review` model class using Django's ORM system. The fields defined correspond to the columns specified in the SQL table definition provided earlier. Note that we use `AutoField` for `review\_id`, as it will automatically assign unique IDs to each instance. We also define foreign key relationships between `Review` instances and `Product` and `Customer` instances using `ForeignKey`. The `\_\_str\_\_` method is overridden to provide a human-readable representation of each `Review` instance.

Here's your requested Django model code:  
  
from django.db import models  
  
class Review(models.Model):  
 review\_id = models.AutoField(primary\_key=True)  
 product\_id = models.ForeignKey('products.Product', on\_delete=models.CASCADE)  
 customer\_id = models.ForeignKey('customers.Customer', on\_delete=models.CASCADE)  
 rating = models.IntegerField()  
 comment = models.TextField()  
 review\_date = models.DateField()  
  
 def \_\_str\_\_(self):  
 return f"Review {self.review\_id} by {self.customer\_id.email}"  
```  
The above code defines a `Review` model class using Django's ORM system. The fields defined correspond to the columns specified in the SQL table definition provided earlier. Note that we use `AutoField` for `review\_id`, as it will automatically assign unique IDs to each instance. We also define foreign key relationships between `Review` instances and `Product` and `Customer` instances using `ForeignKey`. The `\_\_str\_\_` method is overridden to provide a human-readable representation of each `Review` instance.

Here's your requested Django model code:  
  
from django.db import models  
  
class Review(models.Model):  
 review\_id = models.AutoField(primary\_key=True)  
 product\_id = models.ForeignKey('products.Product', on\_delete=models.CASCADE)  
 customer\_id = models.ForeignKey('customers.Customer', on\_delete=models.CASCADE)  
 rating = models.IntegerField()  
 comment = models.TextField()  
 review\_date = models.DateField()  
  
 def \_\_str\_\_(self):  
 return f"Review {self.review\_id} by {self.customer\_id.email}"  
```  
The above code defines a `Review` model class using Django's ORM system. The fields defined correspond to the columns specified in the SQL table definition provided earlier. Note that we use `AutoField` for `review\_id`, as it will automatically assign unique IDs to each instance. We also define foreign key relationships between `Review` instances and `Product` and `Customer` instances using `ForeignKey`. The `\_\_str\_\_` method is overridden to provide a human-readable representation of each `Review` instance.

Here's your requested Django model code:  
  
from django.db import models  
  
class Review(models.Model):  
 review\_id = models.AutoField(primary\_key=True)  
 product\_id = models.ForeignKey('products.Product', on\_delete=models.CASCADE)  
 customer\_id = models.ForeignKey('customers.Customer', on\_delete=models.CASCADE)  
 rating = models.IntegerField()  
 comment = models.TextField()  
 review\_date = models.DateField()  
  
 def \_\_str\_\_(self):  
 return f"Review {self.review\_id} by {self.customer\_id.email}"  
```  
The above code defines a `Review` model class using Django's ORM system. The fields defined correspond to the columns specified in the SQL table definition provided earlier. Note that we use `AutoField` for `review\_id`, as it will automatically assign unique IDs to each instance. We also define foreign key relationships between `Review` instances and `Product` and `Customer` instances using `ForeignKey`. The `\_\_str\_\_` method is overridden to provide a human-readable representation of each `Review` instance.

Here's your requested Django model code:  
  
from django.db import models  
  
class Review(models.Model):  
 review\_id = models.AutoField(primary\_key=True)  
 product\_id = models.ForeignKey('products.Product', on\_delete=models.CASCADE)  
 customer\_id = models.ForeignKey('customers.Customer', on\_delete=models.CASCADE)  
 rating = models.IntegerField()  
 comment = models.TextField()  
 review\_date = models.DateField()  
  
 def \_\_str\_\_(self):  
 return f"Review {self.review\_id} by {self.customer\_id.email}"  
```  
The above code defines a `Review` model class using Django's ORM system. The fields defined correspond to the columns specified in the SQL table definition provided earlier. Note that we use `AutoField` for `review\_id`, as it will automatically assign unique IDs to each instance. We also define foreign key relationships between `Review` instances and `Product` and `Customer` instances using `ForeignKey`. The `\_\_str\_\_` method is overridden to provide a human-readable representation of each `Review` instance.