1.What is Django

Django is a high-level Python web framework that encourages rapid development and clean, pragmatic design. Built by experienced developers, it takes care of much of the hassle of web development, so you can focus on writing your app without needing to reinvent the wheel. It's free and open source.

2.What is the command to install Django and to know about its version?

Django can be installed easily using pip . In the command prompt, execute the following command: pip install django . This will download and install Django. After the installation has completed, you can verify your Django installation by executing django-admin --version in the command prompt.

3.Command to create Django project and app

Set up a Django project	**django-admin startproject <projectname>**
Start a Django app	**python manage.py startapp <appname>**

4.what is the command to run a project in Django?

python manage.py runserver

5.What is the command to create a Superuser in Dango?

$ python manage.py createsuperuser

Username: admin

Email address: admin@example.com

Password: **********

Password (again): *********

Superuser created successfully

6.What is the command for migrations in Django?

$ python manage.py makemigrations

$ python manage.py migrate

7.What is the Django commeand to view a database schema of an existing(or Legacy) Database?

$ python manage.py inspectdb

8. How to view all items in the Model using Django QuerySet?

Models.objects.all()

9. How to filter items in the Model using Django QuerySet?

Models.objects.filter(id=1)

10. How to get a particular item in the Model using Django QuerySet?

Models.objects.get(id=1)

11. How can we combine multiple Queryset in View?

- The Python union operator can be used to combine QuerySets that belong to the same model.

Two or more QuerySets can be combined using the union operator with the following syntax:

```
model_combination = model_set1 | model_set2 | model_set3
```

- You can also use the chain() method from the Itertools module, which allows you to combine two or more QuerySets from different models through concatenation.

```
from itertools import chain

model_combination = list(chain(model_set1, model_set2))
```

12. Explain Django Architecture? Explain Model, View, Template.

Django is based on **MVT (Model-View-Template)** architecture. MVT is a software design pattern for developing a web application.
MVT Structure has the following three parts –
Model: The model is going to act as the interface of your data. It is responsible for maintaining data. It is the logical data structure behind the entire application and is represented by a database (generally relational databases such as MySql, Postgres). To check more, visit – Django Models

View: The View is the user interface — what you see in your browser when you render a website. It is represented by HTML/CSS/Javascript and Jinja files. To check more, visit – Django Views.
Template: A template consists of static parts of the desired HTML output as well as some special syntax describing how dynamic content will be inserted. To check more, visit – Django Templates

13. What is the difference between Project and App?

A Django project is a collection of one or more apps that make up a Web site or other grouping of functionality. A Django app is a component of a project, hopefully encapsulating a logical part of that project, such as a blog app or and app managing file uploads.

So think of a project as like a whole Web site (or mobile app or both) and a Django app as a component of that app, either created by you or a 3rd party app you are integrating in your project.

14. Which is the default database in the settings file in Django?

By default, the configuration uses SQLite.

15. Which is the default port for the Django development server?

By default, the runserver command starts the development server on the internal IP at port 8000.

16. Explain the Migration in Django?

Migrations are Django's way of propagating changes you make to your models (adding a field, deleting a model, etc.) into your database schema. They're designed to be mostly automatic, but you'll need to know when to make migrations, when to run them, and the common problems you might run into.

17. What is Django ORM?

The Django web framework includes a default object-relational mapping layer (ORM) that can be used to interact with data from various relational databases such as SQLite, PostgreSQL, and MySQL. Django allows us to add, delete, modify, and query objects, using an API called ORM. ORM stands for Object Relational Mapping.

18. Explain how can set up the Database in django?

1. Step 1: Create a new project django-admin startproject MyDB.

2. Step 2: Move to the MyDB folder. cd MyDB.
3. Step 3: Create a MySql database.
4. Step 4: Update the settings.py. ...

```
DATABASES = {
   'default': {
      'ENGINE': 'django.db.backends.mysql',
      'NAME': 'mydb',
      'USER': 'root',
      'PASSWORD': 'admin',
      'HOST':'localhost',
      'PORT':'3306',
   }
}
```

5. Step 5: Run the server. ...
6. Step 6: Run the migration command
   python manage.py makemigrations.
   python manage.py migrate.

19. What do you mean by the CSRF token?

A CSRF token is a secure random token (e.g., synchronizer token or challenge token) that is used to prevent CSRF attacks. The token needs to be unique per user session and should be of large random value to make it difficult to guess. A CSRF secure application assigns a unique CSRF token for every user session.

20. What do you mean by the CSRF token?

A QuerySet represents a collection of objects from your database. It can have zero, one or many filters. Filters narrow down the query results based on the given parameters. In SQL terms, a QuerySet equates to a SELECT statement, and a filter is a limiting clause such as WHERE or LIMIT .

21. Which companies use Django?

1. Instagram
2. National Geographic
3. Mozilla
4. Spotify
5. Pinterest

6. Disqus
7. Bitbucket
8. Eventbrite
9. Prezi

22. How static files are defined in Django?Explain Its configuration?

Websites generally need to serve additional files such as images, JavaScript, or CSS. In Django, we refer to these files as "static files". Django provides **django.contrib.staticfiles** to help you manage them.

Configuring static files

1. Make sure that **django.contrib.staticfiles** is included in your **INSTALLED_APPS**.
2. In your settings file, define **STATIC_URL**, for example:

```
STATIC_URL = "static/"
```

3. In your templates, use the **static** template tag to build the URL for the given relative path using the configured **staticfiles STORAGES** alias.

```
{% load static %}
<img src="{% static 'my_app/example.jpg' %}" alt="My image">
```

4. Store your static files in a folder called **static** in your app. For example **my_app/static/my_app/example.jpg**.

Your project will probably also have static assets that aren't tied to a particular app. In addition to using a **static/** directory inside your apps, you can define a list of directories (**STATICFILES_DIRS**) in your settings file where Django will also look for static files. For example:

```
STATICFILES_DIRS = [
    BASE_DIR / "static",
    "/var/www/static/",
]
```