DATABASE INTEGRATION

To integrate a database and perform a query statement using shell scripting, we would typically use a command-line database client.

```
#!/bin/bash

# Database credentials
user="username"
password="password"
host="localhost"
db_name="database_name"

# MySQL command
mysql_cmd="mysql -u $user -p$password -h $host -D $db_name"

# MySQL query
query="SELECT * FROM table_name"

# Execute the command
echo $query | $mysql_cmd
```

API INTEGRATION

What is API?

- ❖ An **Application Programming Interface (API)** is a way for two or more computer programs to communicate with each other.
- ❖ It is a type of software interface, offering a service to other pieces of software.
- ❖ A document or standard that describes how to build or use such a connection or interface is called an API specification.

There are two types of web services used mostly.

- ✓ SOAP – **Simple Object Access Protocol** is a protocol designed to exchange data with the security of programs that are built on different platforms or using different programming languages.

- ✓ REST - **Representational State Transfer** is an API that follows a set of rules through which applications and servers communicate. It was specifically designed for working with components like files, objects, and media components.

SOAP API	REST API
Transports data in standard XML format.	Generally, transports data in JSON. It is based on URI. Because REST follows stateless model, REST does not enforce message format as XML or JSON etc.
Because it is XML based and relies on SOAP, it works with WSDL	It works with GET, POST, PUT, DELETE
Works over HTTP, HTTPS, SMTP	Works over HTTP and HTTPS
Highly structured/typed	Less structured
Designed for large enterprise applications	Designed for mobile device applications

REST API Methods:

Method	Description
GET	Retrieve information about the REST API resource
POST	Create a REST API resource
PATCH	Partially update an existing resource
PUT	Update a REST API resource
DELETE	Delete a REST API resource or related component

Status Codes:

Code range	Category
2xx	Successful Operation
3xx	Redirection
4xx	Client error
5xx	Server error

Code	Meaning	Description
200	OK	The requested action was successful.
201	Created	A new resource was created.
202	Accepted	The request was received, but no modification has been made yet.
204	No Content	The request was successful, but the response has no content.
400	Bad Request	The request was malformed.
401	Unauthorized	The client is not authorized to perform the requested action.
404	Not Found	The requested resource was not found.
415	Unsupported Media Type	The request data format is not supported by the server.
422	Unprocessable Entity	The request data was properly formatted but contained invalid or missing data.
500	Internal Server Error	The server threw an error when processing the request.

CURL COMMAND

curl is a command-line tool used to transfer data to or from a server. It supports a wide range of protocols, including HTTP, HTTPS, FTP, SFTP, SCP, and many more.

With curl, you can make requests to servers, submit form data, upload files, and download files. It's commonly used for interacting with REST APIs, testing server responses, and automating data transfer processes.

Basic Authentication: This is where we provide a username and password which is base64 encoded.

```
curl -u username:password http://example.com
```

Important Options:

-s, --silent: Silent mode. Don't output anything.
-u, --user <user:password>: Set server login details.
-v, --verbose: Make the operation more talkative.
-X, --request <command>: Specify a custom request method to use.

```
#!/bin/bash

# API URL
api_url="https://reqres.in/api/users?page=2"

# Send GET request
response=$(curl -s $api_url)

# Parse the response and extract email and first name
emails=$(echo $response | jq -r '.data[].email')
first_names=$(echo $response | jq -r '.data[].first_name')

# Print the emails and first names
echo "Emails: $emails"
echo "First Names: $first_names"
```

GET

```
#!/bin/bash

# API URL
api_url="https://reqres.in/api/users"

# Credentials
username="your_username"
password="your_password"

# Data
data='{"name": "John", "job": "Developer"}'

# Send POST request
response=$(curl -s -u $username:$password -X POST -d
$data -H "Content-Type: application/json" $api_url)

# Print the response
echo $response
```

POST

EXCEPTION HANDLING

Exception handling in shell scripting is typically done using the trap command, which allows you to catch signals and specify a command or function to be executed when the script receives those signals.

Here's an example of how we might use trap for exception handling in a shell script:

```
#!/bin/bash

# Define a function to execute when an error occurs
handle_error() {
   echo "An error occurred on line $1."
   exit 1
}

# Use trap to specify that the handle_error function should be
run
# when an error occurs (i.e., when any command exits with a
non-zero status)
trap 'handle_error $LINENO' ERR

# The rest of the script...
```