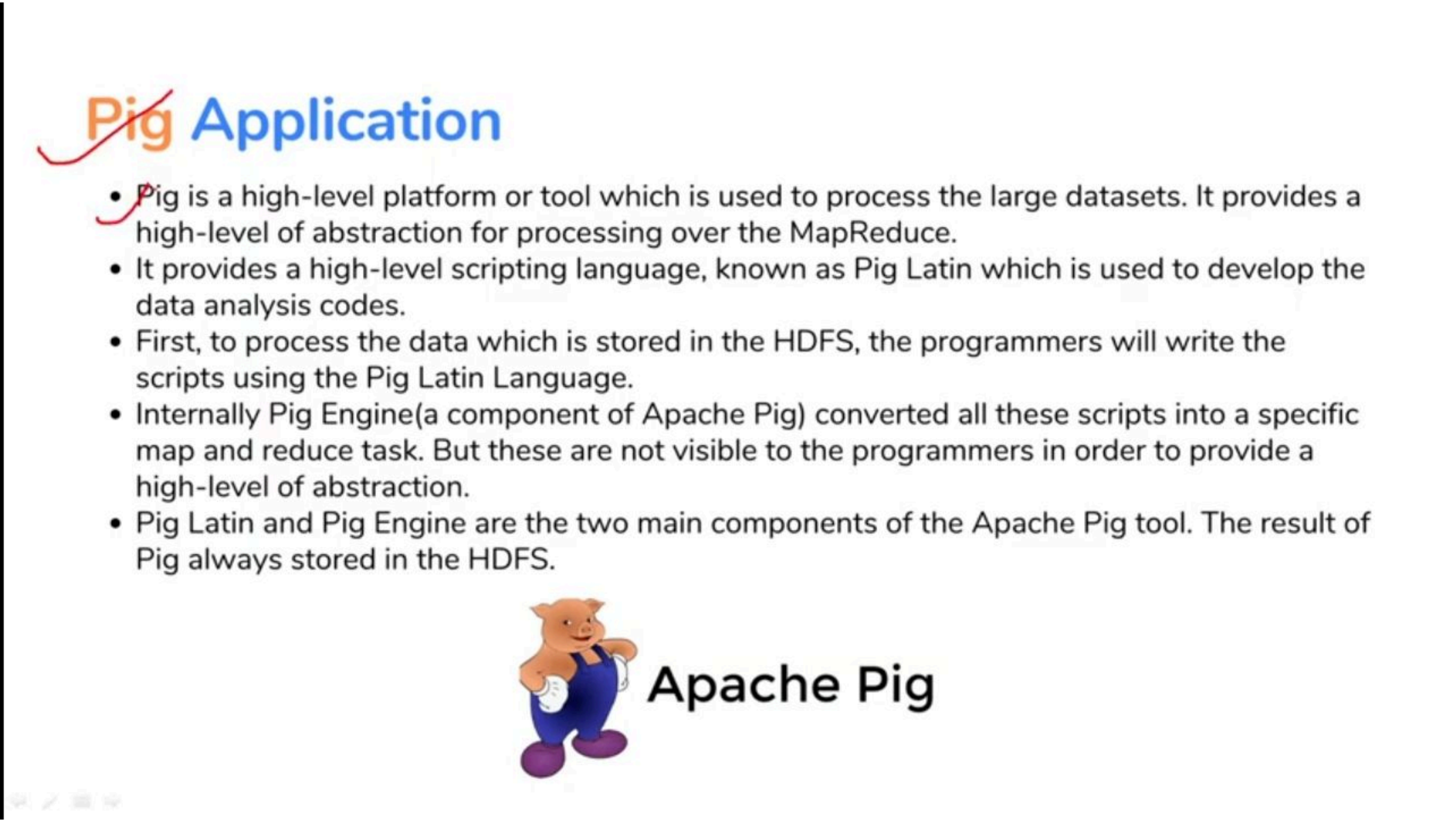
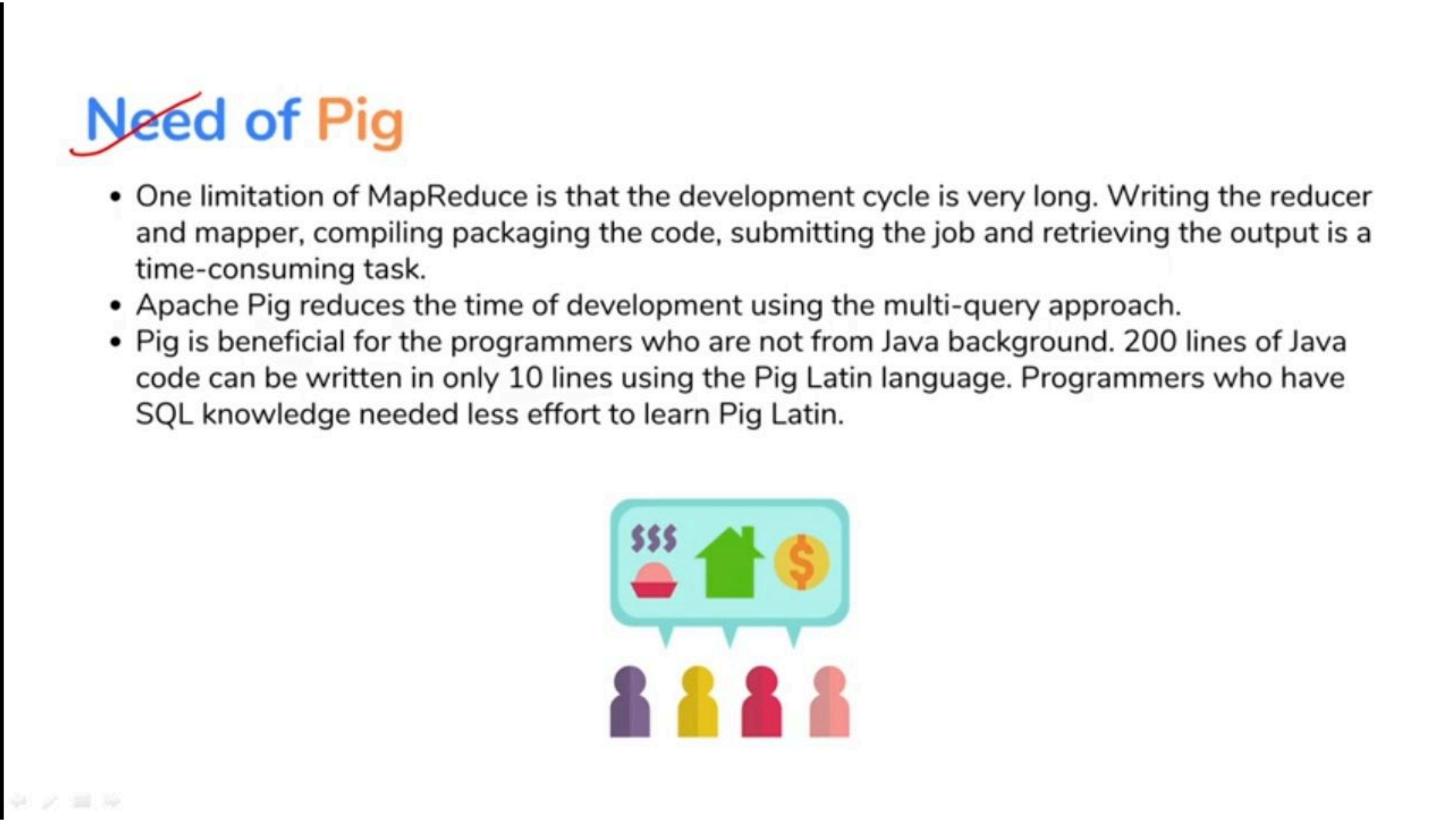


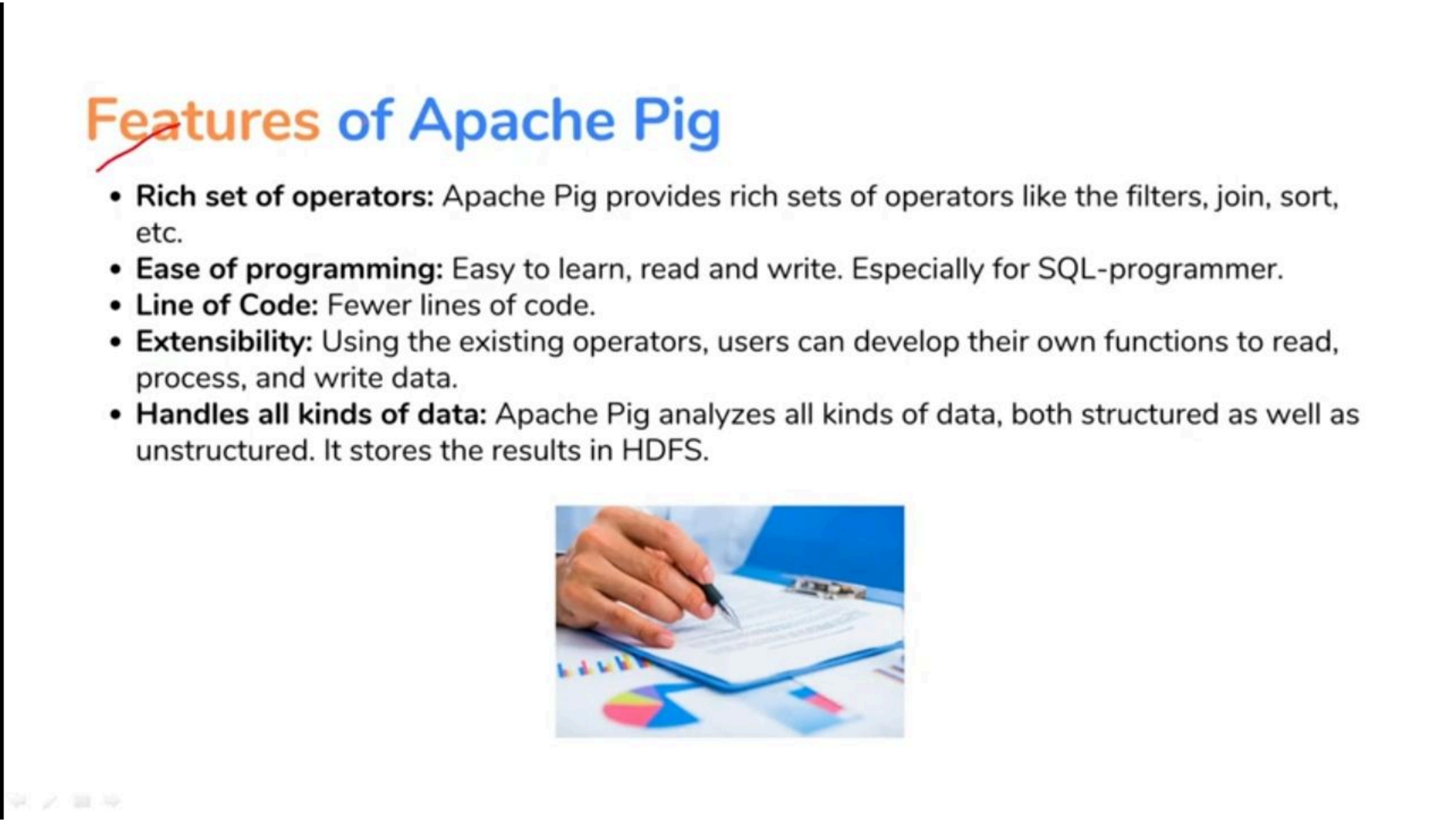
Pig Application

- Pig is a high-level platform or tool which is used to process the large datasets. It provides a high-level of abstraction for processing over the MapReduce.
- It provides a high-level scripting language, known as Pig Latin which is used to develop the data analysis codes.
- First, to process the data which is stored in the HDFS, the programmers will write the scripts using the Pig Latin Language.
- Internally Pig Engine(a component of Apache Pig) converted all these scripts into a specific map and reduce task. But these are not visible to the programmers in order to provide a high-level of abstraction.
- Pig Latin and Pig Engine are the two main components of the Apache Pig tool. The result of Pig always stored in the HDFS.



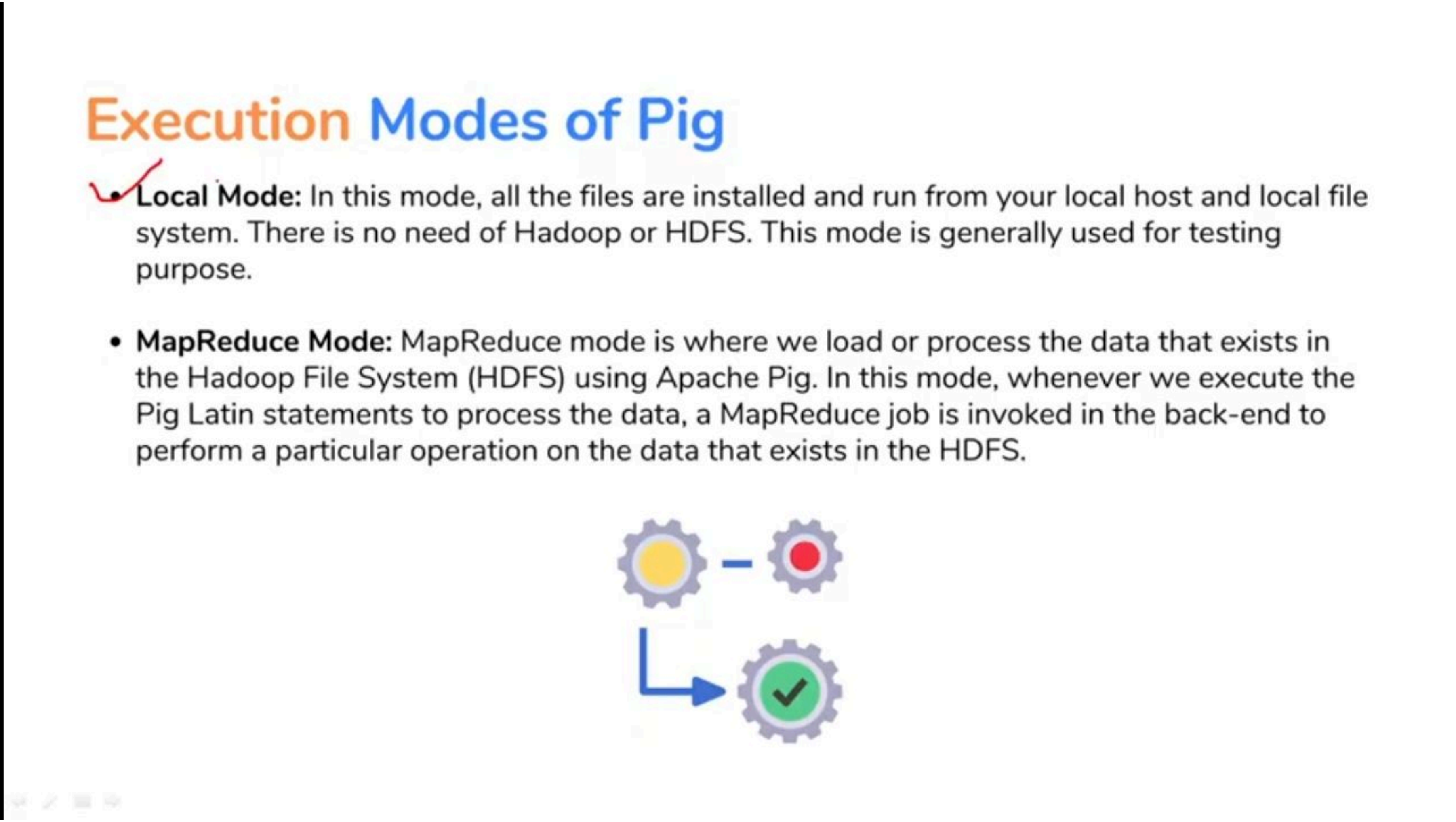
Need of Pig

- One limitation of MapReduce is that the development cycle is very long. Writing the reducer and mapper, compiling packaging the code, submitting the job and retrieving the output is a time-consuming task.
- Apache Pig reduces the time of development using the multi-query approach.
- Pig is beneficial for the programmers who are not from Java background. 200 lines of Java code can be written in only 10 lines using the Pig Latin language. Programmers who have SQL knowledge needed less effort to learn Pig Latin.



Features of Apache Pig

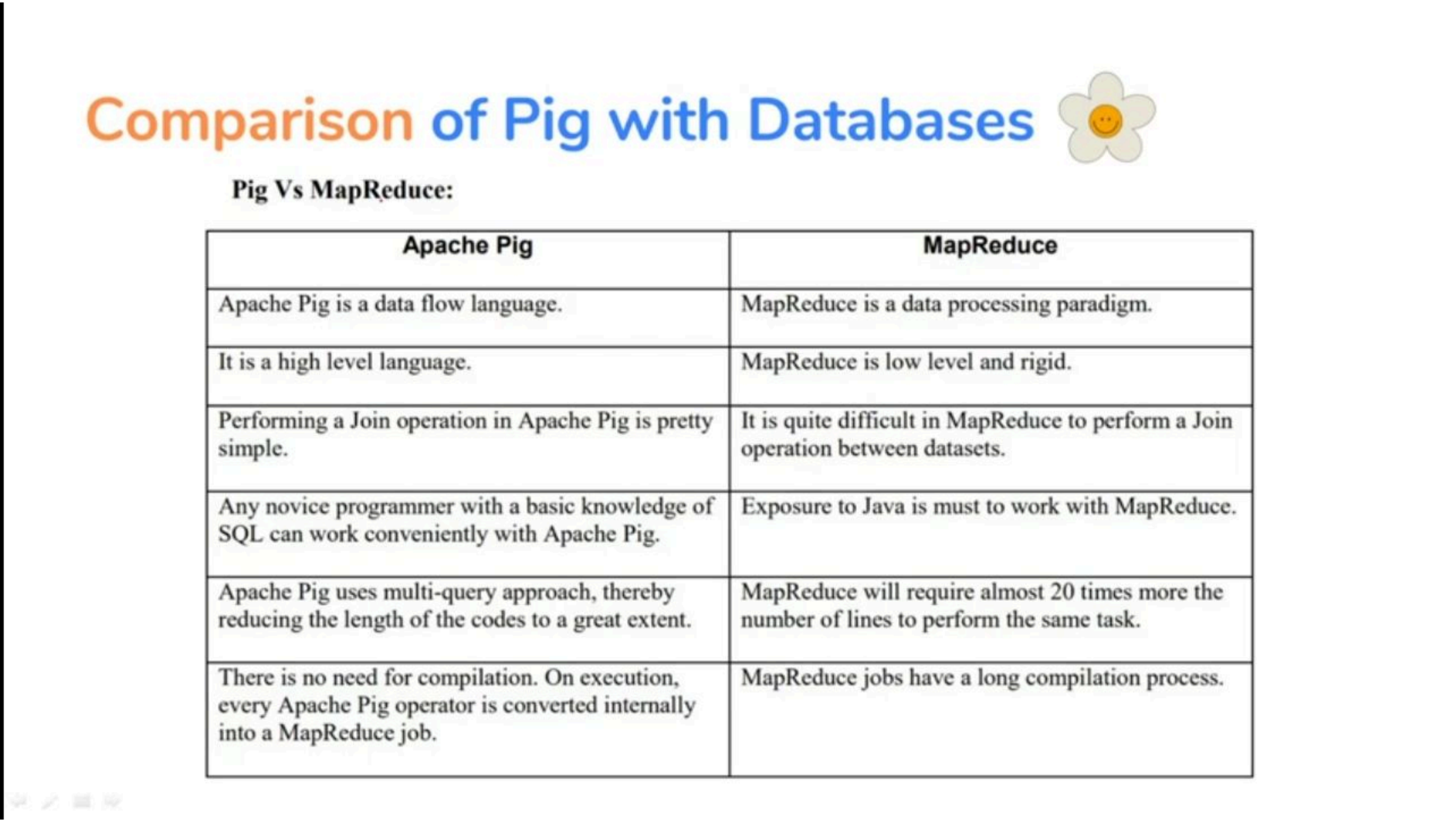
- **Rich set of operators:** Apache Pig provides rich sets of operators like the filters, join, sort, etc.
- **Ease of programming:** Easy to learn, read and write. Especially for SQL-programmer.
- **Line of Code:** Fewer lines of code.
- **Extensibility:** Using the existing operators, users can develop their own functions to read, process, and write data.
- **Handles all kinds of data:** Apache Pig analyzes all kinds of data, both structured as well as unstructured. It stores the results in HDFS.



Execution Modes of Pig

- **Local Mode:** In this mode, all the files are installed and run from your local host and local file system. There is no need of Hadoop or HDFS. This mode is generally used for testing purpose.

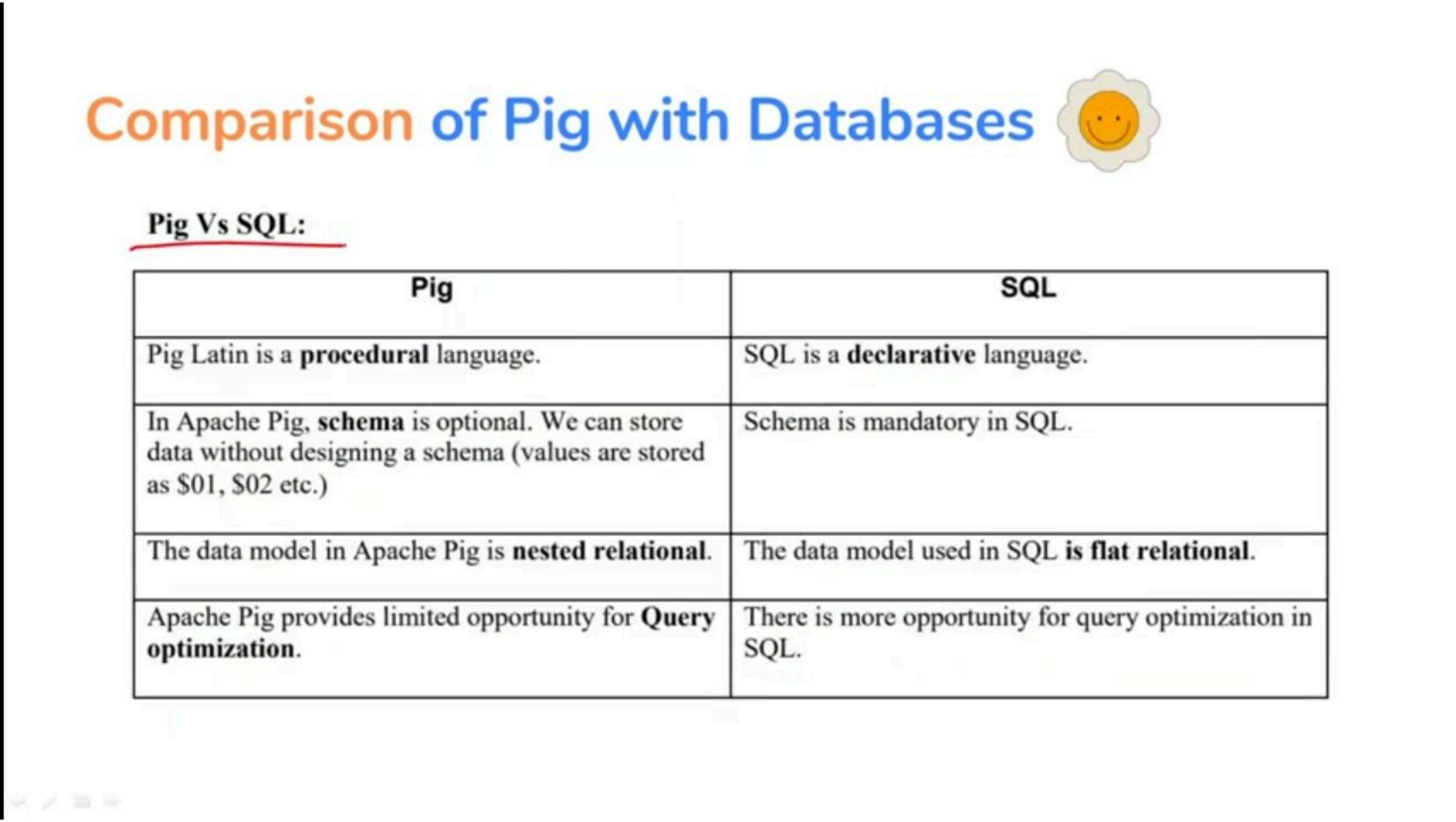
- **MapReduce Mode:** MapReduce mode is where we load or process the data that exists in the Hadoop File System (HDFS) using Apache Pig. In this mode, whenever we execute the Pig Latin statements to process the data, a MapReduce job is invoked in the back-end to perform a particular operation on the data that exists in the HDFS.



Comparison of Pig with Databases

Pig Vs MapReduce:

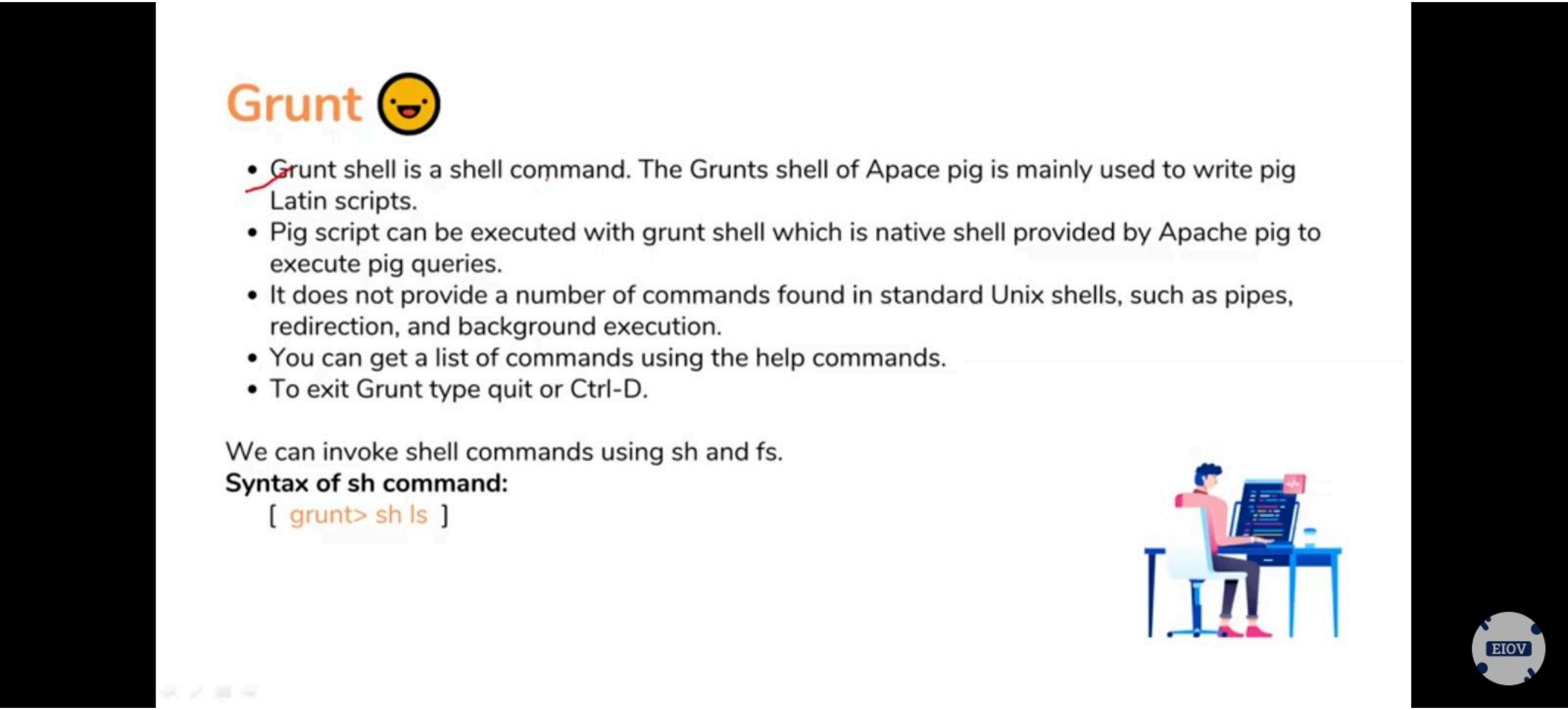
Apache Pig	MapReduce
Apache Pig is a data flow language.	MapReduce is a data processing paradigm.
It is a high level language.	MapReduce is low level and rigid.
Performing a Join operation in Apache Pig is pretty simple.	It is quite difficult in MapReduce to perform a Join operation between datasets.
Any novice programmer with a basic knowledge of SQL can work conveniently with Apache Pig.	Exposure to Java is must to work with MapReduce.
Apache Pig uses multi-query approach, thereby reducing the length of the codes to a great extent.	MapReduce will require almost 20 times more the number of lines to perform the same task.
There is no need for compilation. On execution, every Apache Pig operator is converted internally into a MapReduce job.	MapReduce jobs have a long compilation process.



Comparison of Pig with Databases

Pig Vs SQL:

Pig	SQL
Pig Latin is a **procedural** language.	SQL is a **declarative** language.
In Apache Pig, **schema** is optional. We can store data without designing a schema (values are stored as $01, $02 etc.)	Schema is mandatory in SQL.
The data model in Apache Pig is **nested relational**.	The data model used in SQL **is flat relational**.
Apache Pig provides limited opportunity for **Query optimization**.	There is more opportunity for query optimization in SQL.



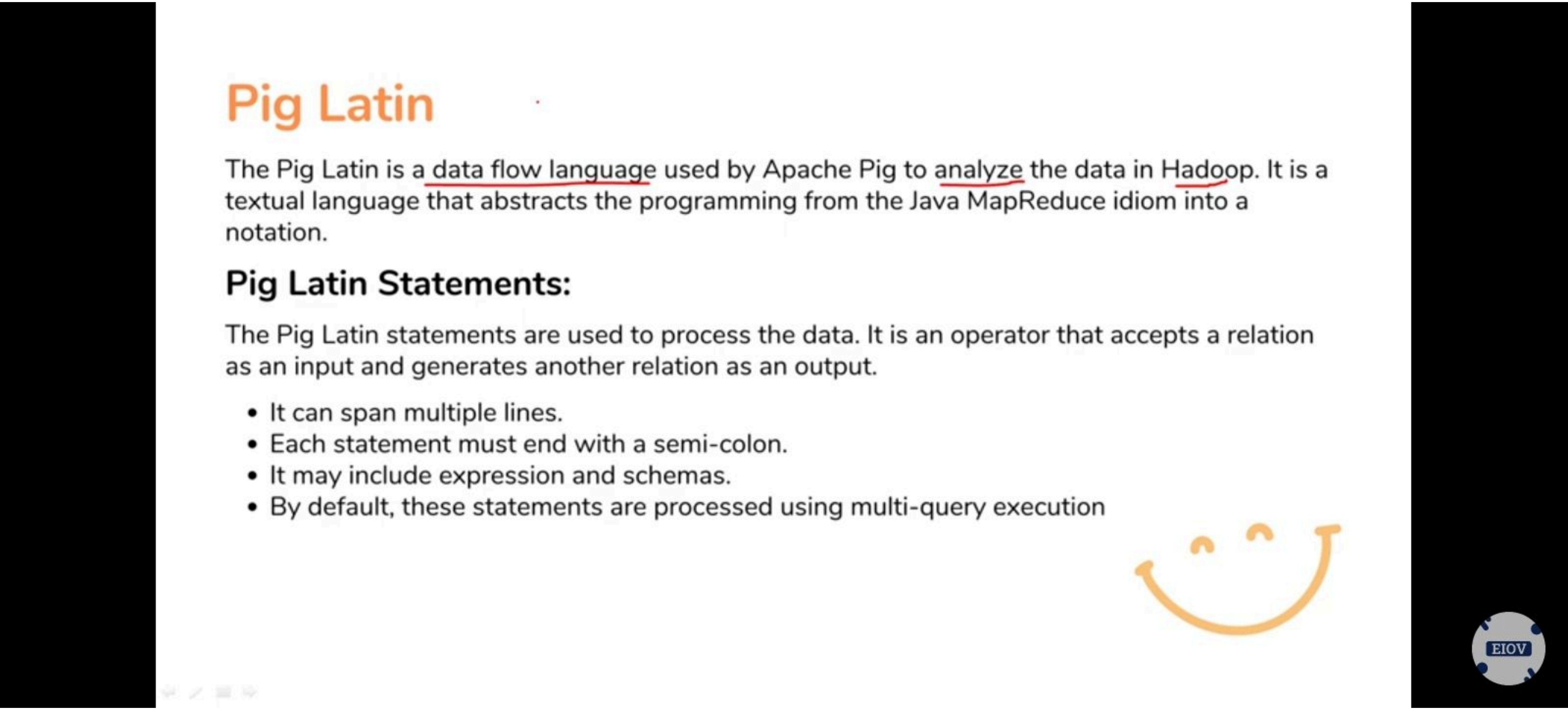
Grunt 😛

- Grunt shell is a shell command. The Grunts shell of Apace pig is mainly used to write pig Latin scripts.
- Pig script can be executed with grunt shell which is native shell provided by Apache pig to execute pig queries.
- It does not provide a number of commands found in standard Unix shells, such as pipes, redirection, and background execution.
- You can get a list of commands using the help commands.
- To exit Grunt type quit or Ctrl-D.

We can invoke shell commands using sh and fs.

Syntax of sh command:

```
[ grunt> sh ls ]
```



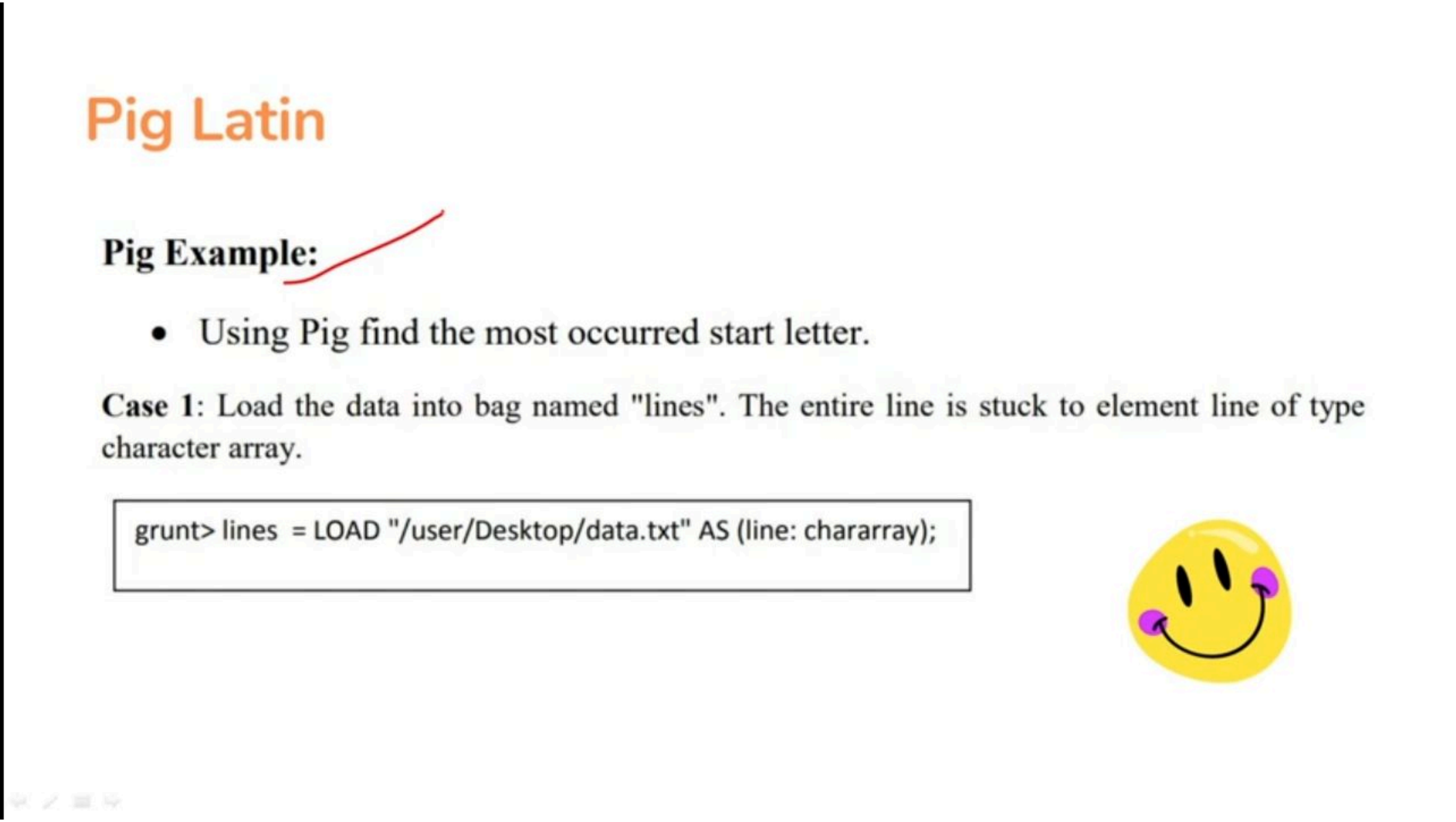
Pig Latin

The Pig Latin is a data flow language used by Apache Pig to analyze the data in Hadoop. It is a textual language that abstracts the programming from the Java MapReduce idiom into a notation.

Pig Latin Statements:

The Pig Latin statements are used to process the data. It is an operator that accepts a relation as an input and generates another relation as an output.

- It can span multiple lines.
- Each statement must end with a semi-colon.
- It may include expression and schemas.
- By default, these statements are processed using multi-query execution



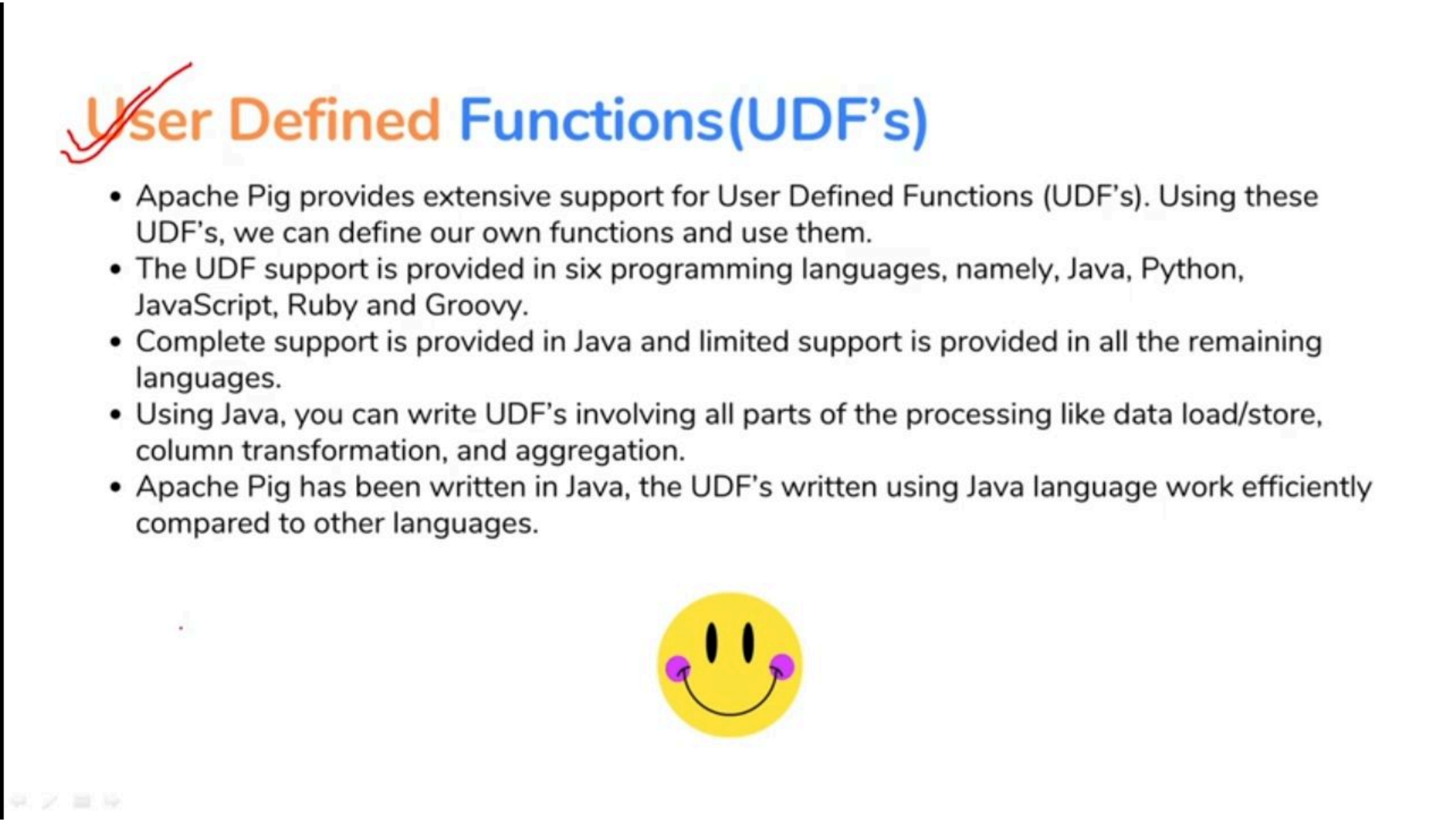
Pig Latin

Pig Example:

- Using Pig find the most occurred start letter.

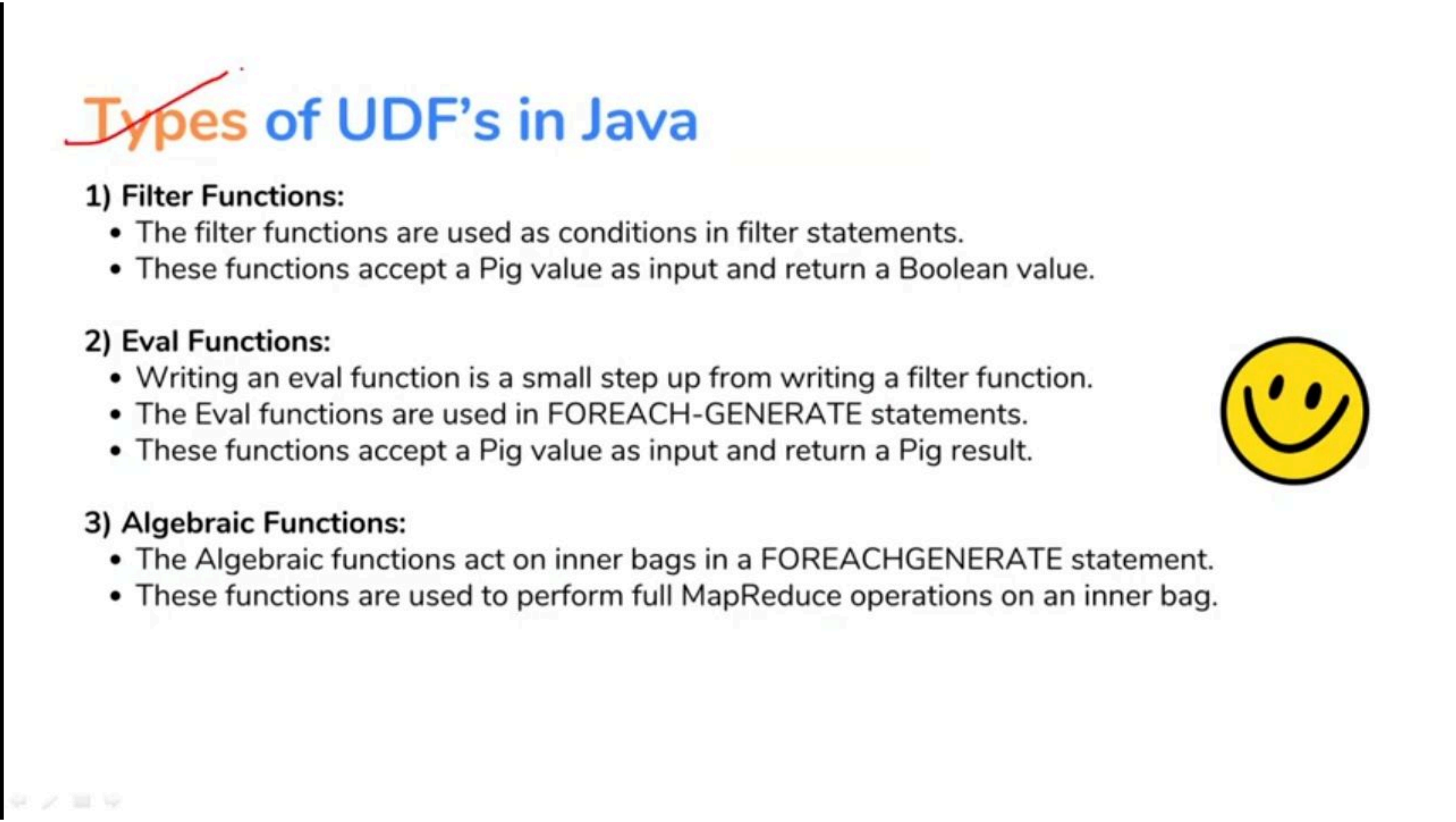
Case 1: Load the data into bag named "lines". The entire line is stuck to element line of type character array.

```
grunt> lines  = LOAD "/user/Desktop/data.txt" AS (line: chararray);
```



User Defined Functions(UDF's)

- Apache Pig provides extensive support for User Defined Functions (UDF's). Using these UDF's, we can define our own functions and use them.
- The UDF support is provided in six programming languages, namely, Java, Python, JavaScript, Ruby and Groovy.
- Complete support is provided in Java and limited support is provided in all the remaining languages.
- Using Java, you can write UDF's involving all parts of the processing like data load/store, column transformation, and aggregation.
- Apache Pig has been written in Java, the UDF's written using Java language work efficiently compared to other languages.



Types of UDF's in Java

1) Filter Functions:

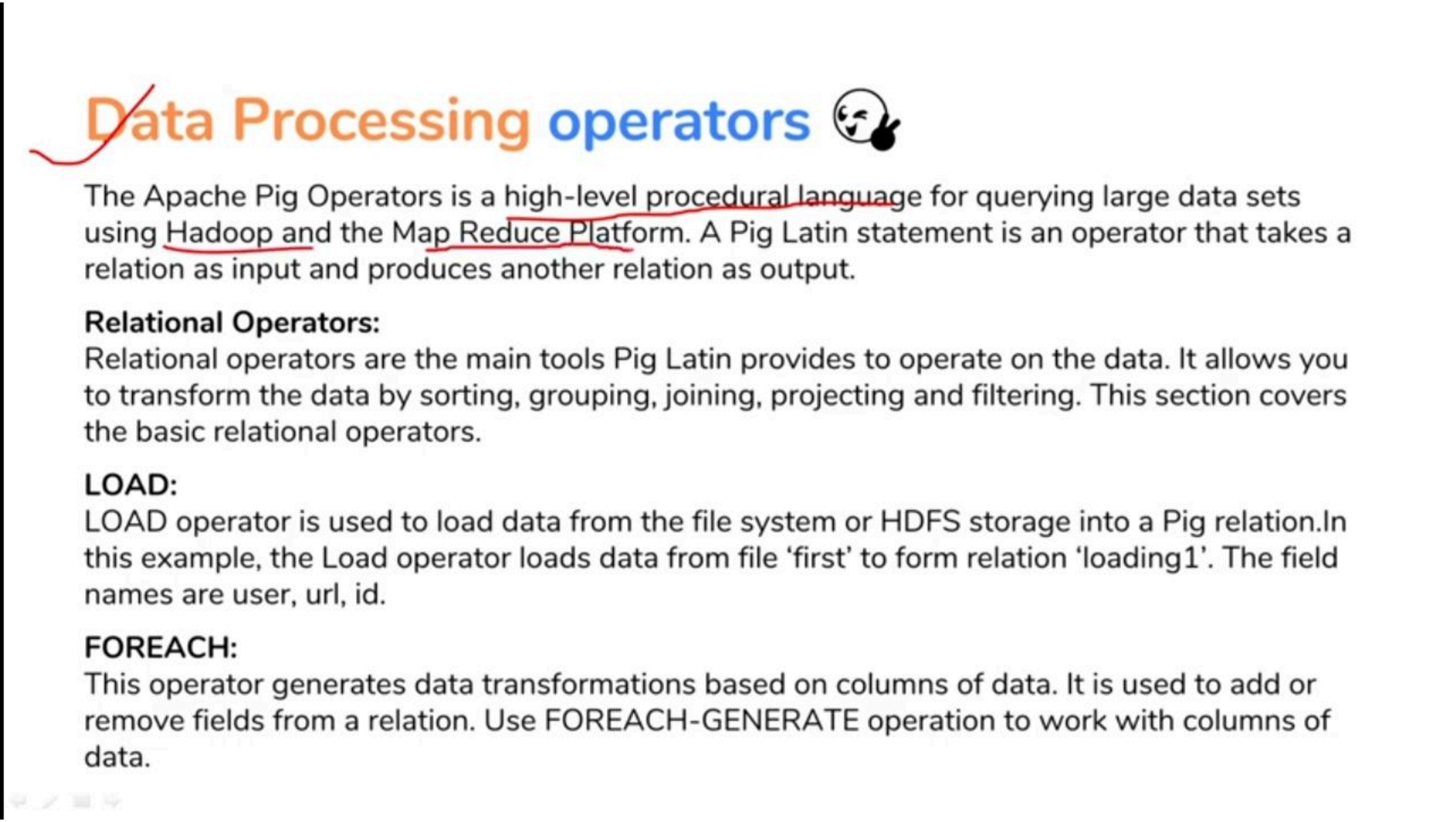
- The filter functions are used as conditions in filter statements.
- These functions accept a Pig value as input and return a Boolean value.

2) Eval Functions:

- Writing an eval function is a small step up from writing a filter function.
- The Eval functions are used in FOREACH-GENERATE statements.
- These functions accept a Pig value as input and return a Pig result.

3) Algebraic Functions:

- The Algebraic functions act on inner bags in a FOREACHGENERATE statement.
- These functions are used to perform full MapReduce operations on an inner bag.



Data Processing operators

The Apache Pig Operators is a high-level procedural language for querying large data sets using Hadoop and the Map Reduce Platform. A Pig Latin statement is an operator that takes a relation as input and produces another relation as output.

Relational Operators:

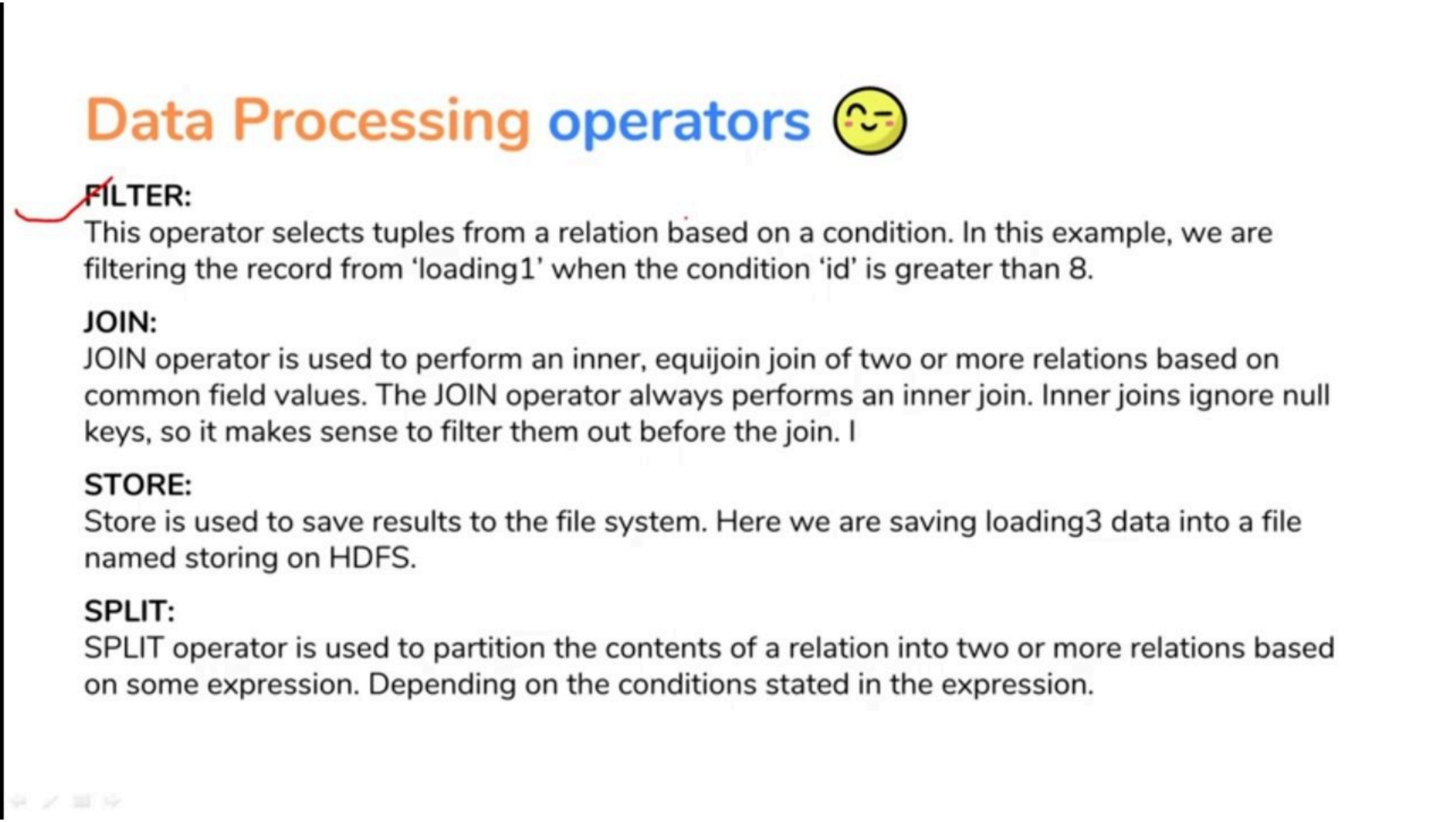
Relational operators are the main tools Pig Latin provides to operate on the data. It allows you to transform the data by sorting, grouping, joining, projecting and filtering. This section covers the basic relational operators.

LOAD:

LOAD operator is used to load data from the file system or HDFS storage into a Pig relation.In this example, the Load operator loads data from file ‘first’ to form relation ‘loading1’. The field names are user, url, id.

FOREACH:

This operator generates data transformations based on columns of data. It is used to add or remove fields from a relation. Use FOREACH-GENERATE operation to work with columns of data.



Data Processing operators

FILTER:
This operator selects tuples from a relation based on a condition. In this example, we are filtering the record from ‘loading1’ when the condition ‘id’ is greater than 8.

JOIN:
JOIN operator is used to perform an inner, equijoin join of two or more relations based on common field values. The JOIN operator always performs an inner join. Inner joins ignore null keys, so it makes sense to filter them out before the join. I

STORE:
Store is used to save results to the file system. Here we are saving loading3 data into a file named storing on HDFS.

SPLIT:
SPLIT operator is used to partition the contents of a relation into two or more relations based on some expression. Depending on the conditions stated in the expression.