**Here are the most important statements IN PIG with examples**:

| **Type** | **Statement** |  | **Example** |
| --- | --- | --- | --- |
| **1. LOAD** | Loads data from a file or database into a relation. | A = LOAD 'student.csv' USING PigStorage(',') AS (id:int, name:chararray, marks:int); |  |
| **2. FOREACH … GENERATE** | Applies a transformation to each record and generates new data. | B = FOREACH A GENERATE name, marks \* 10; |  |
| **3. FILTER** | Filters records based on a condition. | C = FILTER A BY marks > 50; |  |
| **4. GROUP** | Groups records by one or more fields. | D = GROUP A BY name; |  |
| **5. COGROUP** | Groups two or more relations by a common field. | E = COGROUP A BY id, B BY id; |  |
| **6. JOIN** | Joins two or more relations based on a common field. | F = JOIN A BY id, B BY id; |  |
| **7. CROSS** | Returns Cartesian product of two relations. | G = CROSS A, B; |  |
| **8. DISTINCT** | Removes duplicate tuples. | H = DISTINCT A; |  |
| **9. ORDER** | Sorts a relation by one or more fields. | I = ORDER A BY marks DESC; |  |
| **10. LIMIT** | Limits the number of records. | J = LIMIT A 5; |  |
| **11. UNION** | Combines two or more relations with the same schema. | K = UNION A, B; |  |
| **12. SPLIT** | Splits data into two or more relations based on a condition. | SPLIT A INTO high IF marks >= 60, low IF marks < 60; |  |
| **13. STORE** | Stores the output into a file or database. | STORE C INTO 'output' USING PigStorage(','); |  |
| **14. DUMP** | Displays the content of a relation on the console. | DUMP A; |  |

**Example Pig Latin Script**

#**Load student data**

students = LOAD 'student.csv' USING PigStorage(',') AS (id:int, name:chararray, marks:int);

#**Filter students with marks > 50**

passed = FILTER students BY marks > 50;

**#Project name and marks**

names = FOREACH passed GENERATE name, marks;

**#Order by marks descending**

ordered = ORDER names BY marks DESC;

**#Display result**

DUMP ordered;

## Step-by-Step Execution

### 1️⃣ LOAD

students =

(1, Ravi, 75)

(2, Meena, 45)

(3, Arjun, 82)

(4, Sita, 60)

(5, Kiran, 35)

### 2️⃣ FILTER (marks > 50)

passed =

(1, Ravi, 75)

(3, Arjun, 82)

(4, Sita, 60)

### 3️⃣ FOREACH … GENERATE (select only name and marks)

names =

(Ravi, 75)

(Arjun, 82)

(Sita, 60)

### 4️⃣ ORDER (by marks DESC)

ordered =

(Arjun, 82)

(Ravi, 75)

(Sita, 60)

### ✅ Final Output (from DUMP ordered)

(Arjun,82)

(Ravi,75)

(Sita,60)