Exercises

1	2	3	4	5

Surname, First name

Big Data (5294BIDA6Y)
Exam

1	1	1	1	1	1	1	1
2	2	2	2	2	2	2	2
3	3	3	3	3	3	3	3
4	4	4	4	4	4	4	4
5	5	5	5	5	5	5	5
6	6	6	6	6	6	6	6
7	7	7	7	7	7	7	7
8	8	8	8	8	8	8	8
9	9	9	9	9	9	9	9
0	0	0	0	0	0	0	0

This exam has 95 points in total. Out of these, **53 points are needed to pass**.

For multiple choice questions, round boxes indicate that one choice should be made, square boxes indicate that one or more choices should be made.

Foundations of Big Data

1p **1a** List the four V's of Big Data.

1p **1b** Name the three major types of parallelism.

1p **1c** Which scalability type is typically applied for elastic scaling in cloud environments?

(a) Scale-Away
(b) Scale-Out
(c) Scale-Up

1p **1d** Are scalable distributed systems automatically performant?

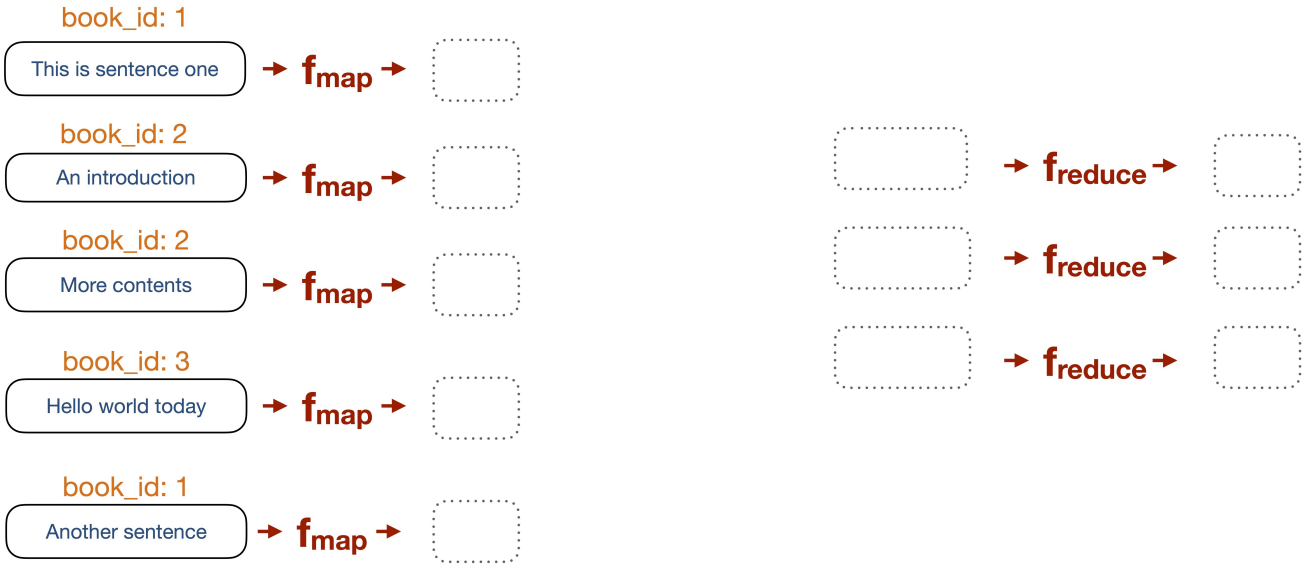
(a) Yes (b) No

1p **1e** Mark all of the following statements that are true.

☐ The cost of storage has decreased enormously in the last decades.
☐ Data scientists spend most of their time designing ML models.
☐ Businesses have to work with more and more less-structured data in recent years.

Distributed Data Processing

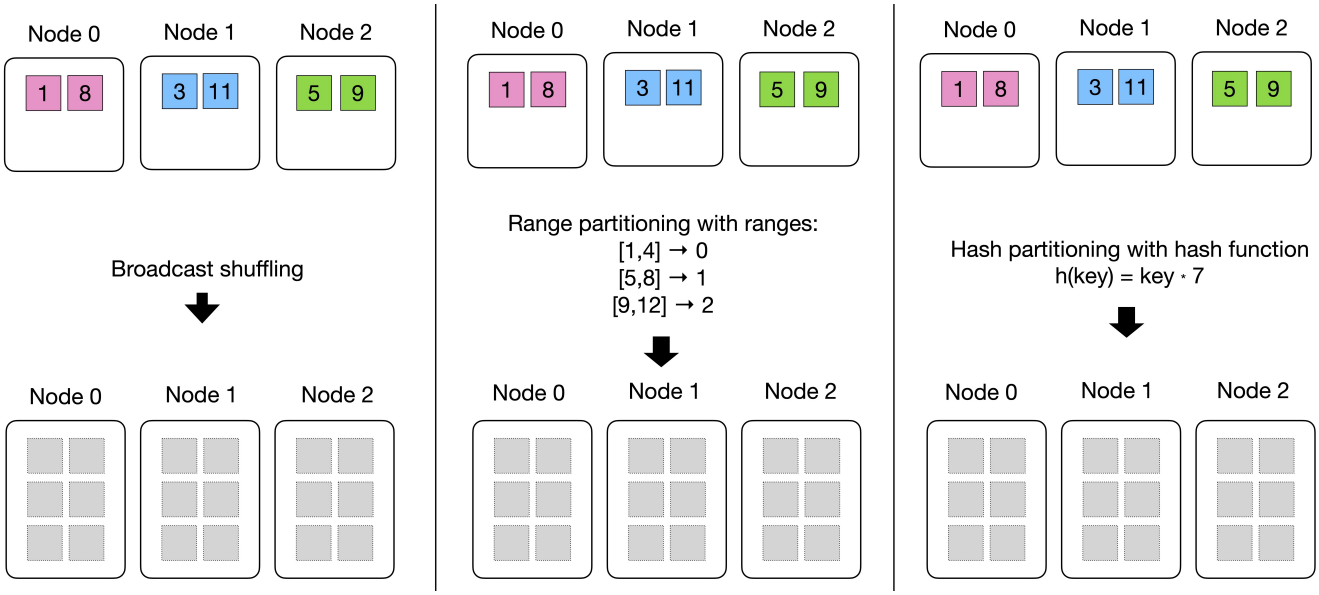
5p **2a** Illustrate how the data moves between the nodes during the distributed execution of the following MapReduce program. **(Draw into the picture)**



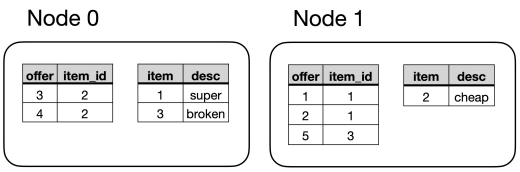

```
fmap (book_id, sentence):
  emit book_id, len(sentence.split(" "))
```

```
freduce (book_id, counts):
  total = 0
  for count in counts:
    total += count
  emit book_id, total
```

5p **2b** Illustrate how the data moves between the nodes in a distributed setup, when we apply: (a) broadcasting (b) range partitioning (c) hash-partitioning. **(Fill the grey boxes in the picture).**

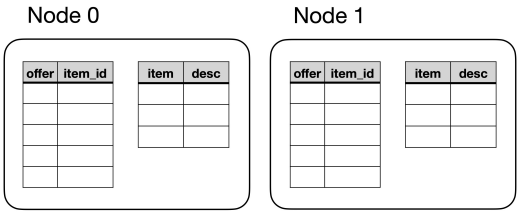


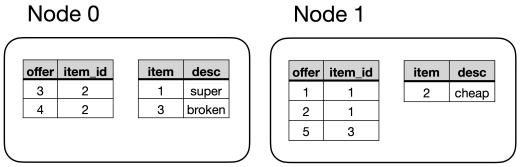
5p **2c** Illustrate whether an asymmetric repartition join or a broadcast join would be a more performant distributed execution strategy for the following join query and partitioned datasets. **(Fill the cells of the partitioned tables for local joining in the bottom part of the picture; quantify the incurred network communication; argue which strategy should be preferred in the box below the diagram).**




```
SELECT *
FROM offers
JOIN items ON item_id = item
```

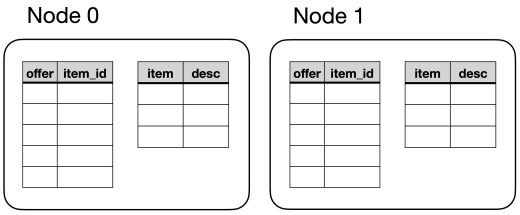
How is the data distributed for local joining with an asymmetric repartition join, using hash partitioning with h(x) = x as hash function?






```
SELECT *
FROM offers
JOIN items ON item_id = item
```

How is the data distributed for local joining with a broadcast join?



Relational Data Processing, MapReduce and Resilient Distributed Datasets

1p **3a** Which of these properties is typically NOT improved by using a distributed database?

(a) System elasticity.
(b) Data consistency.
(c) Fault tolerance.

1p **3b** List two dimensions on which distributed databases are classified.

1p **3c** List the names of the three major phases in which a MapReduce program is executed.

1p **3d** Which of these partitioning techniques is commonly applied in MapReduce engines?

(a) RISC-partitioning
(b) Hash-partitioning
(c) SHA256-partitioning

1p **3e** Identify the correct term: Execution of RDD-based computations with Apache Spark is based on maintaining the ... graph of the RDDs.

(a) lineage
(b) gradient

1p **3f** Mark all common design properties of Mapreduce and Resilient Distributed Datasets.

☐ Programming model based on second-order functions.
☐ Distributed execution on scale-out clusters.
☐ Efficient usage of GPUs.

1p **3g** Which of the following is the major difference between Mapreduce and Resilient Distributed Datasets?

(a) Execution of programs written in Java
(b) Storage of intermediate results in memory
(c) Execution of SQL

1p **3h** To how many partitions of a child RDD does a partition of a parent RDD contribute in general after the application of an operation with narrow dependencies?

(a) None.
(b) One.
(c) Multiple.

1p **3i** To how many partitions of a child RDD does a partition of a parent RDD contribute in general after the application of an operation with wide dependencies?

(a) None.
(b) One.
(c) Multiple.

1p **3j** How do we call the following operations in Apache Spark: count(), collect(), reduce(), save() ?

5p **3k** Identify to which (if any) of the following Spark programs on the left side the SQL queries on the right side are equivalent. Note that spark.sparkContext.broadcast(...) broadcasts data to all worker machines in the cluster. **(Enter Y/N for each box by drawing into the pictures).**

```
orders = spark.read.parquet("gs://path/to/orders").rdd


# PySpark program 1
def order_fn(order):
    if order.priority > 10:
        return [Row(order)]
    else:
        return []

orders.flatMap(lambda order: order_fn(order)) \
  .map(lambda order : Row(id=order.id, supplier=order.supplier))


# PySpark program 2
orders.filter(lambda order: order.priority > 10)


# PySpark program 3
orders
  .map(lambda order => Row(priority=order.priority, \
                           supplier=order.supplier))
```

```
SELECT *
FROM orders
WHERE priority > 10
```

Enter Y/N for each box. This query is equivalent to program ...

☐ 1 ☐ 2 ☐ 3

```
SELECT priority, supplier
FROM orders
WHERE priority > 10
```

Enter Y/N for each box. This query is equivalent to program ...

☐ 1 ☐ 2 ☐ 3

```
orders_df = spark.read.parquet("gs://path/to/orders")
orders = orders_df.rdd
# You can assume that each item_id is unique in the items data
items_df = spark.read.parquet("gs://path/to/items")
items = items_df.rdd


# PySpark program 1
orders_df.join(items_df, on=['item_id']) \
  .groupBy(['item_id']).agg({'price':'sum'})


# PySpark program 2
broadcast_orders = spark.sparkContext.broadcast(orders.collect())

def sum_of_prices_fn(item):
    matched = False
    sum_of_prices = 0.0

    for order in broadcast_orders.value:
        if order.country == "UK":
            if order.item_id == item.item_id:
                matched = True
                sum_of_prices += order.price

    if matched:
        return [Row(item_id=item.item_id, sum_of_prices=sum_of_prices)]
    else:
        return []

items.flatMap(lambda item: sum_of_prices_fn(item)).collect()
```

```
SELECT i.item_id, SUM(o.price)
FROM orders o, items i
WHERE o.country = "UK"
 AND o.item_id <= i.item_id
 AND o.item_id >= i.item_id
GROUP BY i.item_id
```

Enter Y/N for each box. This query is equivalent to program ...

☐ 1 ☐ 2

```
SELECT i.item_id, SUM(o.price)
FROM items i
JOIN orders o ON o.item_id = i.item_id
WHERE o.country = "UK"
GROUP BY i.item_id
```

Enter Y/N for each box. This query is equivalent to program ...

☐ 1 ☐ 2

Data Quality

1p **4a** Which of these business aspects are impacted by data quality?

- [ ] Legal obligations.
- [ ] Operational stability of deployed systems.
- [ ] Predictive quality of ML models.

1p **4b** Which of these are common properties of academic datasets used in ML research?

- [ ] Static data.
- [ ] Thousands of attributes.
- [ ] Fit into memory on a desktop machine.

1p **4c** Which of these is a property of many key-value stores?

(a) Relaxed consistency.
(b) Relaxed durability.
(c) Relaxed atomicity.

1p **4d** List two dimensions of data quality.

1p **4e** What is the foundation of cleaning techniques for quantitative data?

(a) Deduplication.
(b) Referential integrity.
(c) Outlier detection.

5p **4f** Describe the major changes in data collection strategies before and after the internet era.

5p **4g** Employ the "unique row ratio" heuristic to identify the most likely candidate key column in the following table; list intermediate steps and results of the computation and name the identified column.

ssn	year	lastname	OS
1234	1990	Smith	Win
-999	2010	Zhang	Mac
5678	2010	Smith	Mac
-999	2010	Miller	Linux
-999	1999	Miller	Linux
-999	1998	De Vries	Win
-999	2010	Huber	Win
91011	1990	Huber	Win

10p **4h You are given "dirty" data and its corresponding clean ground truth:**

Dirty data:
17 15 25 13 230 11 8

Clean data:
7 8 11 13 15 23 25

(1) Compute the median and 2-trimmed mean in the "dirty" data series; show which of these metrics is closer to its variant on the clean data (to the median and non-trimmend mean in terms of absolute difference);

(2) Illustrate which values from the dirty data are marked as outliers by the interval formed by 1.48 median absolute deviations in the dirty data; list intermediate steps and results of the computation.
Answer both questions in the box below and list intermediate steps.

Implementing Big Data programs

In the next four tasks, we will work with relational data in the following schema:

`Movie(id INT, name VARCHAR, year INT, genre VARCHAR)`

id	name	year	genre
1	The Shawshank Redemption	1994	Drama
2	The Godfather	1972	Thriller
...	...	...	...
8	Pulp Fiction	1994	Thriller
...	...	...	...

`Actor(id INT, firstname VARCHAR, lastname VARCHAR)`

id	firstname	lastname
1	Al	Pacino
...	...	...
22	Uma	Thurman
...	...	...

`PlayedIn(actor_id INT, movie_id INT)`

actor_id	movie_id
1	2
...	...
22	8
...	...

You can assume that there are no two actors with the same first name and last name, and that there are no duplicate entries in PlayedIn.

2p **5a** Write a SQL query to retrieve the id and name of all drama movies released after 1959 and before 1991.

2p **5b** Write a SQL query to count the number of actors in each drama movie

3p **5c** Write a SQL query to find the distinct first name and last name of all actors who played in a movie from the year 1984.

3p **5d** Write a SQL query to find the name of all movies in which both of the actors with id 1 and 22 played.

5p **5e** Implement a mapreduce program (in pseudo-code or python) in form of the functions f_map, f_reduce and f_combine which computes for each day the maximum sensor value measured with a quality level of more than 0.2 from the following data. **Write in to the dashed boxes and also fill in the missing function signatures.**

day	sensor_value	quality
1	100	0.1
1	1000	0.5
...	...	...
5	500	0.0
5	5000	1.0
5	3000	0.8
...	...	...

```
f_map (day, (sensor_value, quality)):
  # YOUR CODE HERE, use emit to emit values
```

```
f_combine (          ,          ):
  # YOUR CODE HERE, use emit to emit values
```

```
f_reduce (          ,          ):
  # YOUR CODE HERE, use emit to emit values
```

5p **5f** Write a SQL query that produces the equivalent result to the following Spark program. (You can assume that there exists an orders and a countries table).

```
countries = {1: 'USA', 2: 'CN', 3:'IN'}

orders = spark.read.parquet("gs://path/to/orders").rdd

def order_fn(order):
    if order.month == "Dec" or order.delay > 10:
        return [Row(order)]
    else:
        return []

filtered_orders = orders.flatMap(lambda order: order_fn(order))


def price_fn(order):
    country_name = countries.get(order.country_id)
    if country_name:
        return [Row((country_name, order.price))]
    else:
        return []

order_prices_by_country_name = filtered_orders \
    .flatMap(lambda order: price_fn(order))

sums = order_prices_by_country_name \
   .reduceByKey(lambda priceA, priceB: priceA + priceB)
```

```
# Write your SQL query here
```

5p **5g** Identify two opportunities to improve the performance of this chain of two MapReduce programs; Rewrite the program (in pseudo-code) to be more performant. **(Write into the dashed box).**

```
f_map1 (movie, visitors_per_week):
  for week, visitors in visitors_per_week:
    if visitors < 0:
      visitors = 0
    emit movie, (week, visitors)

f_reduce1 (movie, weeks_and_visitors):
  for week, visitors in weeks_and_visitors:
    emit movie, (week, visitors)

f_map2 (movie, (week, visitors)):
  if week != 1:
    emit movie, num_visitors

f_reduce2 (movie, visitor_numbers):
  emit movie, max(visitor_numbers)
```

10p **5h** Propose and detail how to efficiently implement a symmetric repartition join in MapReduce; Leverage text, pseudo code and a diagram for your proposal. (You can assume that you can read files from two different locations in the Map-Phase and determine from which location a given input is).