Answers 3.4.

Step 1.

a) Refined Query :

```
SELECT film_id, title

FROM film;
```

b)

Original Query:

```
EXPLAIN

SELECT *

FROM film;
```

Output:

```
Seq Scan on film  (cost=0.00..98.00 rows=1000 width=384)
```

The query scans the entire table and retrieves all columns, with a width=384, indicating the size of data being processed for each row (all columns).

Revised Query:

```
EXPLAIN

SELECT film_id, title

FROM film;
```

Output:

```
Seq Scan on film  (cost=0.00..98.00 rows=1000 width=19)
```

The query still scans the entire table, but only retrieves two columns, with a reduced width=19, indicating much smaller data being processed for each row.

Interpretation:

The cost remains the same for both queries because both perform a sequential scan on the film table. Sequential scans are not affected by the number of columns retrieved; they process all rows in the table. The original query (width=384) processes a large amount of data (all columns), while, the revised query (width=19) processes significantly less data by retrieving only the film_id and title columns. The reduced width makes the revised query more efficient in terms of memory usage and data transfer, even though the cost remains the same.

c)

-Add a LIMIT: If we don't need all rows from the film table, you can limit the number of rows retrieved to optimize performance and reduce processing time:

```
SELECT film_id, title

FROM film

LIMIT 10;
```

This avoids retrieving unnecessary rows and reduces the workload on the database, saving time and resources.

WHERE: If we only need specific rows, you can add a WHERE condition to filter the results:

```
SELECT film_id, title

FROM film

WHERE release_year = 2021;
```

This reduces the number of rows scanned, lowering the query cost and improving efficiency.

Step 2)

The query for sorting the data as requested is:

```
SELECT *

FROM film

ORDER BY title ASC, release_year DESC, rental_rate DESC;
```

Step 3)

a)

```
SELECT rating,

    AVG(rental_rate) AS average_rental_rate

FROM film

GROUP BY rating;
```

b)

```
SELECT rating,

    MIN(rental_duration) AS minimum_rental_duration,

    MAX(rental_duration) AS maximum_rental_duration

FROM film

GROUP BY rating;
```

Step 4)

a)The procedure for migrating data from an external tool to the data warehouse follows the ETL process, which consist in:

1.Extract:

- Collect data from the external source (e.g., the Rockbuster Android app database or API).
- Responsible Team: Data Engineers.
- Tools: Use APIs, connectors, or custom scripts to pull data from the source system.

2.Transform:

- Clean the data to remove inconsistencies (missing values, duplicates).
- Standardize formats (dates, text fields).
- Combine fields or perform calculations if needed (calculate user session duration from timestamps).
- Responsible Team: Data Engineers and Analysts
- Tools: ETL tools like Talend, Apache Nifi, or SQL scripts.

3.Load:

- Insert the transformed data into the data warehouse for storage and analysis.
- Responsible Team: Data Engineers.
- Tools: Database-specific loading tools (cloud platform loaders like AWS Glue).

b) Analyzing data before it has been properly loaded into the data warehouse can lead to several issues:

- Incomplete Data: The dataset might not be fully extracted from the external tool, resulting in partial or incorrect analysis.
- Unclean Data: Without the transformation step, the data may contain errors, inconsistencies, or outliers, leading to misleading insights.
- Performance Issues: Analyzing raw data directly from the external source can strain the source system, slowing down operational processes.
- Lack of Standardization: Data from external tools might not match the formats or structures expected by the analysis tools, making it harder to work with.
- Redundancy or Duplicates: Raw data might contain duplicates or unnecessary fields, complicating the analysis and skewing results.
- Limited Scalability: External systems are not designed to handle large-scale analytical queries, whereas data warehouses are optimized for this purpose.