basic_subquery_operations

```
SELECT m.title, -- Select the movie title
    m.renting_price -- Select the movie's rental price
FROM renting AS r -- Alias the 'renting' table as 'r'
LEFT JOIN movies AS m -- Perform a left join on the 'movies' table aliased as 'm'
ON r.movie_id = m.movie_id; -- Join condition: match movie IDs from both tables
```

Explanation:

- This SQL query retrieves the title and rental price of movies from a database. It uses a LEFT JOIN to combine data from two tables: renting and movies. The join ensures that all rentals are included in the result, even if a corresponding entry is not found in the movies table (in which case the movie title and price would be NULL). The ON clause specifies how the tables are linked—by matching the movie_id column in both tables.

```
SELECT rm.title, -- Report the income from movie rentals for each movie
    SUM(rm.renting_price) AS income_movie
FROM
    (SELECT m.title,
        m.renting_price
    FROM renting AS r
    LEFT JOIN movies AS m
    ON r.movie_id=m.movie_id) AS rm
GROUP BY rm.title
ORDER BY income_movie ASC; -- Order the result by increasing income
```

Explanation:

- This SQL query calculates the total rental income for each movie. It uses a subquery (rm) to join the renting and movies tables, then aggregates the results using SUM() to find the total income for each movie title. Finally, it orders the results in ascending order of income.

```
SELECT
  a.gender,  -- Report for male and female actors from the USA
  MAX(a.year_of_birth),  -- The year of birth of the oldest actor
  MIN(a.year_of_birth)  -- The year of birth of the youngest actor
FROM
  (SELECT *
   FROM actors
   WHERE nationality = 'USA'
  ) AS a  -- Use a subsequent SELECT to get all information about actors from the USA. Give the
```

```
table the name 'a'
GROUP BY
  a.gender;
```

Explanation:

This SQL query finds the oldest and youngest actors from the USA, categorized by gender. It first selects all actors with nationality ‘USA’ using a subquery. Then, it groups the results by gender and uses aggregate functions MAX() and MIN() to determine the maximum and minimum birth years within each gender group.