

Reviews by year.**User**

If I have these tables:

```

CREATE TABLE Customer
(asin_id VARCHAR (255),
reviewer_id VARCHAR (255),
reviewer_name VARCHAR (255),
PRIMARY KEY (reviewer_id, asin_id));

CREATE TABLE Review (
reviewer_id VARCHAR(255),
asin_id VARCHAR(255),
reviewer_name VARCHAR(255),
helpful_numerator INT,
helpful_denominator INT,
review TEXT,
summary TEXT,
overall DECIMAL(2,1) CHECK (overall >= 0 AND overall <= 5),
unix_review_time BIGINT,
review_time DATE,
PRIMARY KEY (asin_id, reviewer_id));

CREATE TABLE Appliance
(asin_id VARCHAR (255) PRIMARY
KEY, title VARCHAR (255),
category VARCHAR (255),
description TEXT, brand VARCHAR
(255), price DECIMAL (10,3),
imURL TEXT, related TEXT,
salesrank INTEGER);

```

How would I make this sql query "How many reviews submitted every January for years: 2011, 2012, 2013, 2014?"

ChatGPT

To answer the query "How many reviews were submitted every January for the years 2011, 2012, 2013, and 2014?" using the provided tables, you can write an SQL query that filters the reviews by the specified years and month. The `Review` table contains a `review_time` column of type DATE, which you can use for this purpose. Here is the SQL query:

```

```sql
SELECT
 EXTRACT(YEAR FROM review_time) AS year,
 COUNT(*) AS review_count
FROM
 Review
WHERE
 EXTRACT(MONTH FROM review_time) = 1
 AND EXTRACT(YEAR FROM review_time) IN (2011, 2012, 2013, 2014)
GROUP BY
 EXTRACT(YEAR FROM
review_time) ORDER BY year;
```

```

Explanation:

- `EXTRACT(YEAR FROM review_time)`: Extracts the year part from the `review_time` column.
- `EXTRACT(MONTH FROM review_time)`: Extracts the month part from the `review_time` column.
- The `WHERE` clause filters the reviews to include only those from January (where the month equals 1) and from the years 2011, 2012, 2013, and 2014.
- `GROUP BY EXTRACT(YEAR FROM review_time)`: Groups the results by year.
- `COUNT(*) AS review_count`: Counts the number of reviews for each year.- `ORDER BY year`: Orders the results by year. This query will return the number of reviews submitted in January for each of the specified years.

User

What if I wanted the reviews submitted for every subcategory for the month of may cross years 2011-2014?

ChatGPT

To get the reviews submitted for every subcategory in the month of May across the years 2011 to 2014, you can join the `Review` table with the `Appliance` table on `asin_id`, filter by the month of May, and the years 2011 to 2014, then group by the subcategory (category) and the year. Here's the SQL query:

```

```sql
SELECT
 a.category AS subcategory,
 EXTRACT(YEAR FROM r.review_time) AS year,
 COUNT(*) AS review_count
FROM
 Review r
JOIN
 Appliance a ON r.asin_id = a.asin_id
WHERE
 EXTRACT(MONTH FROM r.review_time) = 5
 AND EXTRACT(YEAR FROM r.review_time) IN (2011, 2012, 2013, 2014)
GROUP BY a.category,
 EXTRACT(YEAR FROM r.review_time)
ORDER BY
 a.category,
 year;
```

```

Explanation:

- ``JOIN Appliance a ON r.asin_id = a.asin_id``: Joins the ``Review`` table with the ``Appliance`` table using the ``asin_id`` field.
- ``EXTRACT(MONTH FROM r.review_time) = 5``: Filters the reviews to include only those from May (where the month equals 5).
- ``EXTRACT(YEAR FROM r.review_time) IN (2011, 2012, 2013, 2014)``: Filters the reviews to include only those from the years 2011 to 2014.
- ``GROUP BY a.category, EXTRACT(YEAR FROM r.review_time)``: Groups the results by subcategory and year.
- ``COUNT(*) AS review_count``: Counts the number of reviews for each subcategory and year.- ``ORDER BY a.category, year``: Orders the results by subcategory and year. This query will return the number of reviews submitted in May for each subcategory (category) and each year from 2011 to 2014.

User

Can you create a query that would find the average price of every product subcategory that has an overall rating greater than 3?

ChatGPT

To find the average price of every product subcategory where the products have an overall rating greater than 3, you need to join the ``Review`` table with the ``Appliance`` table, filter the reviews with an overall rating greater than 3, and then calculate the average price for each subcategory. Here is the SQL query:

```
```sql
SELECT
 a.category AS subcategory,
 AVG(a.price) AS average_price
FROM
 Review r
JOIN
 Appliance a ON r.asin_id = a.asin_id
WHERE
 r.overall > 3
GROUP BY
 a.category
ORDER BY
 a.category;
```
```

Explanation:

- ``JOIN Appliance a ON r.asin_id = a.asin_id``: Joins the ``Review`` table with the ``Appliance`` table using the ``asin_id`` field.
- ``WHERE r.overall > 3``: Filters the reviews to include only those with an overall rating greater than 3.
- ``GROUP BY a.category``: Groups the results by subcategory.
- ``AVG(a.price) AS average_price``: Calculates the average price for each subcategory.- ``ORDER BY a.category``: Orders the results by subcategory.

This query will return the average price of products for each subcategory where the products have an overall rating greater than 3.