1-

C .

Before Update

rental_id	film_id	customer_rate
10310	2	50
13421	2	50
7733	2	50
15218	2	50
4364	2	50
10992	2	50
11758	2	50

film_id	score
2	(Null)

After Update

rental_id	film_id	customer_rate
10310	2	50
13421	2	50
7733	2	50
15218	2	50
4364	2	69
10992	2	50
11758	2	50

film_id	score
2	52.00

-2

A . زمانی که یک تریگر، به گونه ای کار کند که دوباره خودش فعال شود (به طور مثال جدولی را تغییر دهد که تغییر دادن آن باعث فعال شدن تریگر میشود)، به صورت بازگشتی بار ها و بار ها اجرا میشود و میتواند مشکلات زیادی به وجود اورد، این محدودیت جلوی این اتفاق را میگیرد

B . با استفاده از تابع pg_trigger_depth . این تابع عمق تریگر های صدا زده شده را برمیگرداند، یعنی اگر در اولین اجرا باشد مقدار بازگشتی این تابع 0 است

-3

```
select rent_count from customer where customer_id = 1;
insert into rental(rental_date,inventory_id,customer_id,return_date,staff_id,last_update) values(NOW(),170,1,NULL,1,NOW());
insert into rental(rental_date,inventory_id,customer_id,return_date,staff_id,last_update) values(NOW(),174,1,NULL,1,NOW());
insert into rental(rental_date,inventory_id,customer_id,return_date,staff_id,last_update) values(NOW(),189,1,NULL,1,NOW());
insert into rental(rental_date,inventory_id,customer_id,return_date,staff_id,last_update) values(NOW(),196,1,NULL,1,NOW());
select rent_count from customer where customer_id = 1;
insert into rental(rental_date,inventory_id,customer_id,return_date,staff_id,last_update) values(NOW(),201,1,NULL,1,NOW());
select rent_count from customer where customer_id = 1;
select rental_id,film_id,category_id,rental_date from rental join inventory using(inventory_id) join film using (film_id) join film_category using(film_id) where
customer_id = 1 order by rental_date desc;
```

Message | Result 1 | Result 2 | Result 3 | Result 4

rent_count
0

Message | Result 1 | Result 2 | Result 3 | Result 4

rent_count
4

Message | Result 1 | Result 2 | Result 3 | Result 4

rent_count
0

rental_id	film_id	category_id	rental_date
16068	428	8	2021-12-29 16:29:48.494536
16067	45	13	2021-12-29 16:29:48.494536
16066	44	14	2021-12-29 16:29:48.491515
16065	43	8	2021-12-29 16:29:48.490206
16064	39	14	2021-12-29 16:29:48.488678
16063	37	4	2021-12-29 16:29:48.482367

-4

film_title	film_rating	rank_in_all	rank_in_rating	sum_amount	is_in_first_quartile
Academy Dinosaur	PG	718	147	33.79	No
Ace Goldfinger	G	508	89	52.93	No
Adaptation Holes	NC-17	699	137	34.89	No
Affair Prejudice	G	261	47	83.79	No
African Egg	G	557	99	47.89	No
Agent Truman	PG	118	24	111.81	Yes
Airplane Sierra	PG-13	263	64	82.85	No
Airport Pollock	NC-17	239	50	86.85	Yes
Alabama Devil	PG-13	354	80	71.88	No
Aladdin Calendar	NC-17	66	18	131.77	Yes
Alamo Videotape	G	761	132	30.78	No
Alaska Phantom	PG	675	135	37.77	No
Ali Forever	PG	493	101	54.91	No
Alien Center	NC-17	251	52	84.80	No
Alley Evolution	NC-17	568	108	46.87	No
Alone Trip	R	433	85	61.83	No
Alter Victory	PG-13	759	172	30.80	No
Amadeus Holy	PG	731	149	32.80	No
Amelie Hellfighters	R	381	72	67.90	No
American Circus	R	31	5	146.81	Yes
Amistad Midsummer	G	367	67	70.79	No
Anaconda Confessions	R	525	106	51.82	No
Analyze Hoosiers	R	487	96	55.86	No

---- Q4 with sum_amounts as (select film_id,sum(amount) as sum_amount from payment join rental using(rental_id) join inventory using(inventory_ | Read Only | Query time: 0.049s | Record 1 of 958

month_number	rating	sum_amount	prev_month_sales	next_month_sales
2	G	1422.60	(Null)	3944.62
2	PG	1658.99	(Null)	4757.52
2	PG-13	1856.58	(Null)	5316.63
2	R	1740.79	(Null)	4772.77
2	NC-17	1672.88	(Null)	5095.02
3	G	3944.62	1422.60	5040.03
3	PG	4757.52	1658.99	5725.45
3	PG-13	5316.63	1856.58	6563.76
3	R	4772.77	1740.79	5455.79
3	NC-17	5095.02	1672.88	5774.43
4	G	5040.03	3944.62	104.63
4	PG	5725.45	4757.52	94.69
4	PG-13	6563.76	5316.63	118.59
4	R	5455.79	4772.77	82.71
4	NC-17	5774.43	5095.02	113.56
5	G	104.63	5040.03	(Null)

---- Q5 select extract(month from payment_date) as month_number,rating, sum(amount) as sum_amount, lag(sum(amount),1) over (p | Read Only | Query time: 0.048s | Record 1 of 20

-6

. A

```
139  ---- Q6
140  ---A
141  select film.film_id,staff.staff_id,store.store_id,address.city_id,
142  count(rental.rental_id) as total_rent
143  from rental, staff, inventory,film,store,address
144  where rental.inventory_id = inventory.inventory_id and
145  inventory.film_id = film.film_id and
146  rental.staff_id = staff.staff_id and
```

Message | Result 1 | Result 2

film_id	staff_id	store_id	city_id	total_rent
(Null)	(Null)	(Null)	(Null)	16050
(Null)	1	1	300	8046
(Null)	1	(Null)	(Null)	8046
(Null)	(Null)	1	300	8046
(Null)	(Null)	1	(Null)	8046
(Null)	(Null)	(Null)	300	8046
(Null)	1	1	(Null)	8046
(Null)	1	(Null)	300	8046
(Null)	2	2	(Null)	8004
(Null)	(Null)	2	(Null)	8004
(Null)	2	(Null)	(Null)	8004
(Null)	(Null)	2	576	8004
(Null)	(Null)	(Null)	576	8004
(Null)	2	(Null)	576	8004
(Null)	2	2	576	8004
103	(Null)	(Null)	(Null)	34

---B select film.film_id,staff.staff_id,store.store_id,address.city_id, count(rental.rental_id) as total_rent from rental, staff, inventory,film,st | Read Only | Query time: 0.233s | Record 1 of 14357

. B

```
154  ---B
155  select film.film_id,staff.staff_id,store.store_id,address.city_id,
156  count(rental.rental_id) as total_rent
157  from rental, staff, inventory,film,store,address
158  where rental.inventory_id = inventory.inventory_id and
159  inventory.film_id = film.film_id and
160  rental.staff_id = staff.staff_id and
161  staff.store_id = store.store_id and
```

Message | Result 1

film_id	staff_id	store_id	city_id	total_rent
(Null)	(Null)	(Null)	(Null)	16050
(Null)	1	1	300	8046
(Null)	1	(Null)	(Null)	8046
(Null)	(Null)	1	300	8046
(Null)	(Null)	1	(Null)	8046
(Null)	(Null)	(Null)	300	8046
(Null)	1	1	(Null)	8046
(Null)	1	(Null)	300	8046
(Null)	2	2	(Null)	8004
(Null)	(Null)	2	(Null)	8004
(Null)	2	(Null)	(Null)	8004
(Null)	(Null)	2	576	8004
(Null)	(Null)	(Null)	576	8004
(Null)	2	(Null)	576	8004
(Null)	2	2	576	8004
103	(Null)	(Null)	(Null)	34

--B select film.film_id,staff.staff_id,store.store_id,address.city_id, count(rental.rental_id) as total_rent from rental, staff, inventory,film,st | Read Only | Query time: 0.145s | Record 1 of 14357

-7

A . connecting to data , preparing application to receive data , fetching data into application , displaying data , (editing, validating and saving data if needed)

B .

ODBC	**OLEDB**
Originally designed for relational databases. (Since changed)	Originally designed for non-relational and relational databases.
On-going support for SQL	SQL support void 2019
Component-based	Procedural-based
More difficult to deploy	Easier to deploy

C . ORM یا Object-Relational Mapping در یک دیتابیس relational اجازه انجام کوئری و تغییر دیتا را به صورت شی گرا میدهد

مزایا و فواید: سیستم مدیریت دیتابیس را abstract میکند و منعطف تر است

این مدل به صورت خیلی ضعیف به دیگر قسمت های اپلیکیشن متصل است، به همین علت میتوان آن را راحت در صورت نیاز تغییر داد

امکان استفاده از مزایای برنامه نویسی شی گرا (مانند وراثت) را به راحتی به برنامه نویس میدهد

با روش طبیعی کدنویسی برنامه نویس مطابقت دارد (چون از خود زبان است)

نمونه های ORM:

Hibernate for Java

Propel for PHP

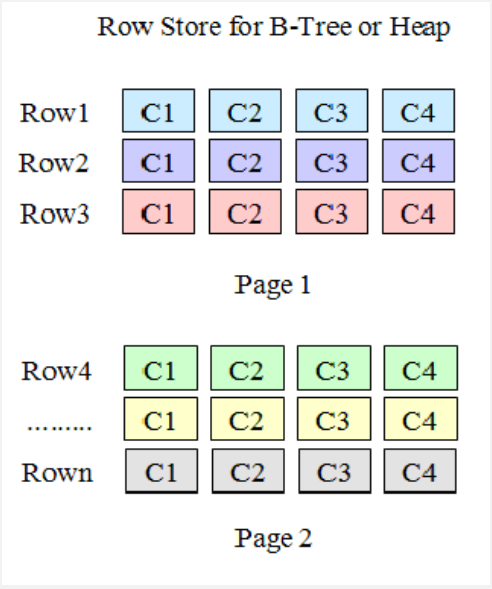
Django for Python

Entity Framework for C#

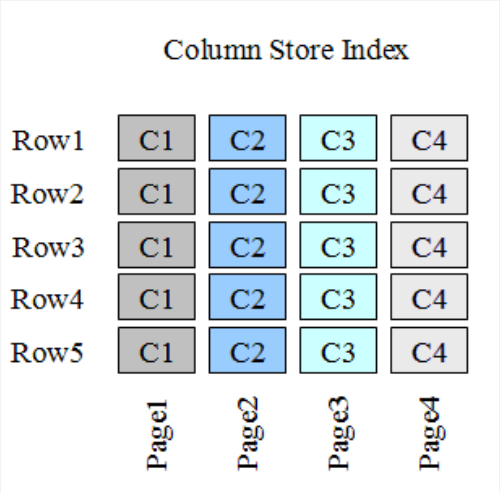
-8

. A

در روش ذخیره سازی Row Store، مقادیر ستونها در یک سطر بصورت متوالی ذخیره می‌شوند، در این روش ذخیره سازی از ساختار B-Tree یا Heap استفاده می‌شود.



در ذخیره سازی Column Store، داده‌ها بصورت ستونی ذخیره می‌شوند،در این روش داده ها، فشرده سازی می‌شوند و اینکار باعث می‌شود،در زمان درخواست یک Query، نیاز به Disk I/o به حداقل برسد، در نتیجه، زمان و سرعت پاسخگویی به پرس و جوها بسیار افزایش می‌یابد.



B . سرعت پایین کوئری های OLAP ، حجم بیشتر نسبت به columnstore (به دلیل عدم فشرده سازی)

C . columnstore، زیرا دیتا های زیاد را میتواند با سرعت بیشتری به دست اورد و برای عملیات read-only و decision support مناسب تر است.

D . rowstore، زیرا برای عملیات های نوشتن و transaction processing مناسب تر است و columnstore هزینه های بیشتری در این نوع عملیات ها دارد.

E .