SELECT \* FROM movie;

SELECT distinct country FROM movie;

SELECT distinct country FROM movie ORDER BY country ASC;

SELECT distinct country FROM movie WHERE country IS NOT NULL ORDER BY country ASC;

SELECT COUNT(DISTINCT country) FROM movie;

SELECT

COUNT(\*) as total,

COUNT(\*) - COUNT(birthdate) as birth\_null,

COUNT(birthdate) as birth\_not\_null

FROM person;

SELECT \* FROM movie

WHERE 1=1

AND release\_year>=2000

AND release\_year<=2010;

SELECT \*

FROM movie

WHERE 1=0

OR release\_year=2000

OR release\_year=2010;

SELECT \*

FROM movie

WHERE release\_year BETWEEN 2000 AND 2010

AND country = 'USA';

SELECT \*

FROM movie

WHERE release\_year IN (2000, 2010);

SELECT \* FROM movie WHERE release\_year LIKE '200\_';

SELECT \* FROM movie WHERE release\_year LIKE '2%';

SELECT AVG(budget) FROM movie;

SELECT MAX(budget) FROM movie;

SELECT SUM(budget) FROM movie;

SELECT

AVG(budget) as avg,

MAX(budget) as max,

SUM(budget) as sum

FROM movie;

SELECT (4 \* 3);

SELECT title

FROM movie

ORDER BY release\_year DESC;

SELECT birthdate, name

FROM person

ORDER BY birthdate, name;

select release\_year, GROUP\_CONCAT(CONCAT('{id:"', id, '", title:"',title,'"}')) as list from movie group by release\_year;

SET group\_concat\_max\_len = 1000000;

SELECT m.title, COUNT(\*) as cnt

FROM role r

JOIN movie m on r.movie\_id=m.id

GROUP BY m.id

HAVING cnt >= 3;

1/ Зачем нужны индексы?

Производительность. Unique, primary, foreign keys.

2/ BTREE индекс. Понятие.

3/ Когда применяется. Найти строки с key=10, key>10, 5<key<10

4/ Индекс по символьному полю. Collation. Like.

5/ Индекс по нескольким колонкам. Склеиваются, создается один индекс.

6/ Индексы дорогие.

И на запись, - обновление индексов

и на чтение. – Больше размер индекса, меньше влазит в оперативную память, больше операций ввода-вывода. Мускл оптимизатор дольше анализирует какой индекс использовать для текущего запроса.

alter table movie add key (country, language, release\_year);

alter table movie add key (release\_year, country, language);

explain select \* from movie where release\_year>2010 and country='USA' and language='English';

explain select \* from movie where release\_year>2011;

explain select \* from movie where release\_year>2012;

7/ Covering index

alter table movie add key (country, release\_year);

explain select country, min(release\_year) from movie group by country;

8/ Индексы и джоины

Нестед лупс

Все джоины должны быть проиндексированы.

Длина индекса

explain SELECT movie.title, COUNT(\*) as cnt FROM role JOIN movie on role.movie\_id=movie.id GROUP BY movie.id HAVING cnt >= 3;

4954 rows in set (0.13 sec)

alter table role add key (movie\_id);

SELECT movie.title, COUNT(\*) as cnt FROM role JOIN movie on role.movie\_id=movie.id JOIN GROUP BY movie.id HAVING cnt >= 3

SELECT movie.\*, COUNT(movie.id) as people\_cnt FROM role JOIN movie on role.movie\_id=movie.id JOIN review on review.movie\_id=movie.id GROUP BY movie.id HAVING people\_cnt >= 3;

4954 rows in set (5.20 sec)

explain

SELECT movie.title, COUNT(movie.id) as people\_cnt, GROUP\_CONCAT(person.name)

FROM role

JOIN movie on role.movie\_id=movie.id

JOIN review on review.movie\_id=movie.id

JOIN person ON person.id=role.person\_id

GROUP BY movie.id HAVING people\_cnt >= 3;