

Урок 3. Работа с данными через Hive и Hue. Обзор PIG и Impala

Выполнил Колеганов Н.Д.

Задание

[исследовательское задание] Разобраться с тем, что такое bucketing.

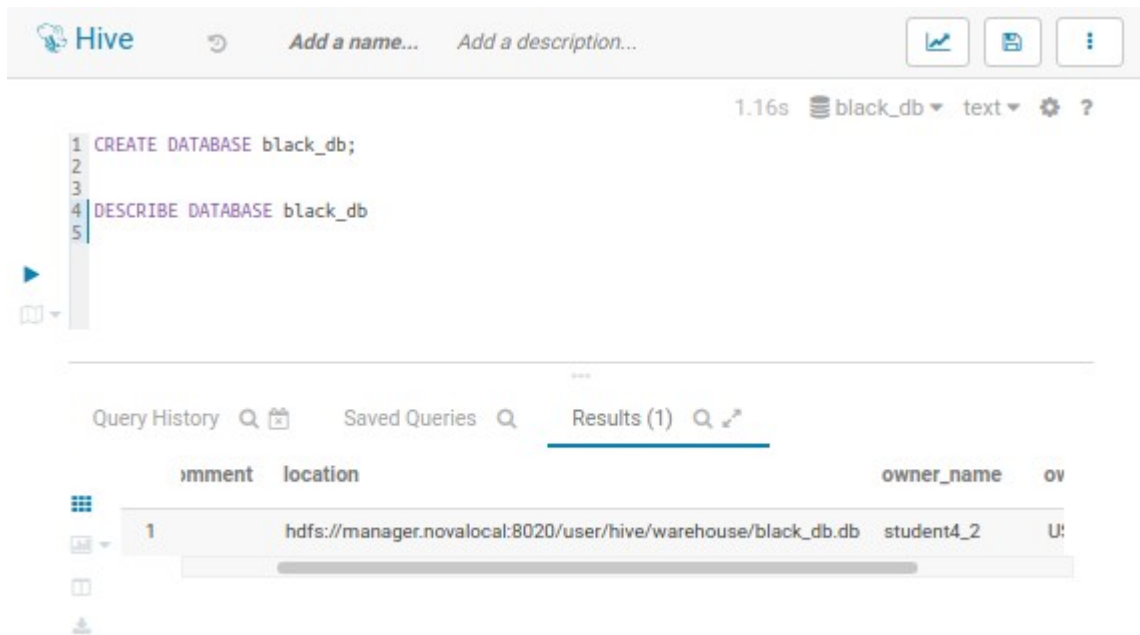
1. Скачать любой датасет из списка ниже <https://www.kaggle.com/shuyangli94/food-com-recipes-and-user-interactions> <https://www.kaggle.com/datasnaek/youtube-new> <https://www.kaggle.com/akhilv11/border-crossing-entry-data> <https://www.kaggle.com/tristan581/17k-apple-app-store-strategy-games> <https://www.kaggle.com/gustavomodelli/forest-fires-in-brazil>
2. Загрузить этот датасет в HDFS в свою домашнюю папку
3. Создать собственную базу данных в HIVE
4. Создать EXTERNAL таблицы внутри базы данных с использованием всех загруженных файлов. Один файл – одна таблица.
5. Сделать любой запрос по загруженным данным используя групповые и агрегатные функции.
6. Сделать любой запрос по загруженным данным используя JOIN.
7. [Продвинутый вариант] Сделать все вышеперечисленное с использованием JSON SerDe. Подсказка: см в сторону команды «ADD JAR» Если последнее задание выполнено -- то пункты 2-5 не обязательно делать.

Решение

Bucketing - это метод оптимизации, который использует **сегменты** (и **столбцы**), чтобы определить разбиение данных и избежать перемешивания данных.

Мотивация состоит в том, чтобы оптимизировать производительность запроса на соединение, избегая перемешивания (или *обмена*) таблиц, участвующих в соединении. Bucketing приводит к меньшему количеству обменов (этапов).

Создал БД black_db



Создал таблицу forest

```
create table black_db.forest
```

```
(
```

```
year_    int,
```

```
state    string,
```

```
month_    string,
```

```
number    int,
```

```
date_    string
```

```
)
```

```
ROW FORMAT SERDE
```

```
    'org.apache.hadoop.hive.serde2.OpenCSVSerde'
```

```
STORED AS INPUTFORMAT
```

'org.apache.hadoop.mapred.TextInputFormat'

OUTPUTFORMAT

'org.apache.hadoop.hive ql.io.HiveIgnoreKeyTextOutputFormat'

TBLPROPERTIES (

'serialization.null.format' = '',

'skip.header.line.count' = '1')

;

```
7 create table black_db.forest
8 (
9   year_      int,
10  state      string,
11  month_     string,
12  number     int,
13  date_      string
14 )
15 ROW FORMAT SERDE
16   'org.apache.hadoop.hive.serde2.OpenCSVSerde'
17 STORED AS INPUTFORMAT
18   'org.apache.hadoop.mapred.TextInputFormat'
19 OUTPUTFORMAT
20   'org.apache.hadoop.hive ql.io.HiveIgnoreKeyTextOutputFormat'
21 TBLPROPERTIES (
22   'serialization.null.format' = '',
23   'skip.header.line.count' = '1')
24 ;
25
```

✓ Success.

Query History Saved Queries

2 минуты назад ✓

```
create table black_db.forest ( year_int, state string,
month_string, number int, date_string ) ROW FORMAT SERDE
'org.apache.hadoop.hive.serde2.OpenCSVSerde' STORED AS INPUTFORMAT
'org.apache.hadoop.mapred.TextInputFormat' OUTPUTFORMAT
'org.apache.hadoop.hive ql.io.HiveIgnoreKeyTextOutputFormat'
TBLPROPERTIES ( 'serialization.null.format' = '',
'skip.header.line.count' = '1')
```

LOAD DATA INPATH '/user/student4_2/amazon.csv' INTO TABLE black_db.forest

;

```
25
26
27 LOAD DATA INPATH '/user/student4_2/amazon.csv' INTO TABLE black_db.forest
28 ;
```

✓ Success.

Query History Saved Queries

минуту назад ✓

```
LOAD DATA INPATH '/user/student4_2/amazon.csv' INTO TABLE
black_db.forest
```

black_db.forest

	forest_year_	forest.state	forest.month_	forest.number	forest.date_
1	1998	Acre	Janeiro	0	1998-01-01
2	1999	Acre	Janeiro	0	1999-01-01
3	2000	Acre	Janeiro	0	2000-01-01
4	2001	Acre	Janeiro	0	2001-01-01
5	2002	Acre	Janeiro	0	2002-01-01
6	2003	Acre	Janeiro	10	2003-01-01
7	2004	Acre	Janeiro	0	2004-01-01
8	2005	Acre	Janeiro	12	2005-01-01
9	2006	Acre	Janeiro	4	2006-01-01
10	2007	Acre	Janeiro	0	2007-01-01
11	2008	Acre	Janeiro	0	2008-01-01
12	2009	Acre	Janeiro	0	2009-01-01
13	2010	Acre	Janeiro	1	2010-01-01
14	2011	Acre	Janeiro	0	2011-01-01
15	2012	Acre	Janeiro	0	2012-01-01

Accel Tabla Browser

Создал таблицу appstore

```
create table black_db.appstore
(
  URL      string,
  ID       int,
  Name     string,
  Subtitle int,
  IconURL  string,
  AvrUsrRate float,
  UsrRateCount float,
  Price    float,
  InAppPurchase string,
  Description string,
  Developer string,
  AgeRate  string,
  Lang     string,
  Size_    int,
  PrimGenre string,
  Genres   string,
  OriginalRelease string,
  CurRelease string
)
ROW FORMAT SERDE
  'org.apache.hadoop.hive.serde2.OpenCSVSerde'
STORED AS INPUTFORMAT
  'org.apache.hadoop.mapred.TextInputFormat'
OUTPUTFORMAT
  'org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat'
TBLPROPERTIES (
  'serialization.null.format' = '',
  'skip.header.line.count' = '1'
);

LOAD DATA INPATH '/user/student4_2/appstore_games.csv' INTO TABLE black_db.appstore
;
```

The screenshot displays a Hive query execution interface. At the top, there are tabs for "Query History" and "Saved Queries". Below these, a query execution log shows a successful execution of a query. The query text is displayed in a monospaced font, showing the creation of the 'black_db.appstore' table and the loading of data from '/user/student4_2/appstore_games.csv' into the table. The execution status is "Success." and the time taken is "несколько секунд назад" (a few seconds ago).

```
create table black_db.appstore ( URL string, ID int, Name string,
Subtitle int, IconURL string, AvrUsrRate float, UsrRateCount float,
Price float, InAppPurchase string, Description string,
Developer string, AgeRate string, Lang string, Size_ int,
PrimGenre string, Genres string, OriginalRelease string,
CurRelease string ) ROW FORMAT SERDE
'org.apache.hadoop.hive.serde2.OpenCSVSerde' STORED AS INPUTFORMAT
'org.apache.hadoop.mapred.TextInputFormat' OUTPUTFORMAT
'org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat'
TBLPROPERTIES ( 'serialization.null.format' = '',
'skip.header.line.count' = '1')

LOAD DATA INPATH '/user/student4_2/appstore_games.csv' INTO TABLE black_db.appstore
;
```

Success.

Query History Saved Queries

несколько секунд назад

```
LOAD DATA INPATH '/user/student4_2/appstore_games.csv' INTO
TABLE black_db.appstore
```

Columns Details Sample Analysis

	appstore.url	appstore.id	appstore.name
1	https://apps.apple.com/us/app/sudoku/id284921427	284921427	Sudoku
2	https://apps.apple.com/us/app/reversi/id284926400	284926400	Reversi
3	https://apps.apple.com/us/app/morocco/id284946595	284946595	Morocco
4	https://apps.apple.com/us/app/sudoku-free/id285755462	285755462	Sudoku (Free)
5	https://apps.apple.com/us/app/senet-deluxe/id285831220	285831220	Senet Deluxe
6	https://apps.apple.com/us/app/sudoku-classic-number-puzzle/id286210009	286210009	Sudoku - Classic number puzzl
7	https://apps.apple.com/us/app/gravitation/id286313771	286313771	Gravitation
8	https://apps.apple.com/us/app/colony/id286363959	286363959	Colony
9	https://apps.apple.com/us/app/carte/id286566987	286566987	Carte
10	https://apps.apple.com/us/app/barrels-o-fun/id286682679	286682679	'Barrels O' Fun'
11	https://apps.apple.com/us/app/quaddraxx/id287563734	287563734	Quaddraxx
12	https://apps.apple.com/us/app/lumen-lite/id288096268	288096268	Lumen Lite
13	https://apps.apple.com/us/app/bubblepop/id288669794	288669794	BubblePop
14	https://apps.apple.com/us/app/marple/id288689440	288689440	Marple
15	https://apps.apple.com/us/app/tetravex-lite/id288918962	288918962	Tetravex Lite

Создал таблицу border

```
create table black_db.border
(
  PortName string,
  State string,
  PortCode int,
  Border string,
  Date_ string,
  Value int,
  Location_ string )
ROW FORMAT SERDE
  'org.apache.hadoop.hive.serde2.OpenCSVSerde'
STORED AS INPUTFORMAT
  'org.apache.hadoop.mapred.TextInputFormat'
OUTPUTFORMAT
  'org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat'
TBLPROPERTIES (
  'serialization.null.format' = '',
  'skip.header.line.count' = '1');
```

```
67
68 create table black_db.border
69 (
70 PortName    string,
71 State       string,
72 PortCode    int,
73 Border      string,
74 Date_       string,
75 Value       int,
76 Location_   string )
77 ROW FORMAT SERDE
78 'org.apache.hadoop.hive.serde2.OpenCSVSerde'
79 STORED AS INPUTFORMAT
80 'org.apache.hadoop.mapred.TextInputFormat'
81 OUTPUTFORMAT
82 'org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat'
83 TBLPROPERTIES (
84 'serialization.null.format' = '',
85 'skip.header.line.count' = '1');
```

✓ Success.

Query History Saved Queries

несколько секунд назад

```
create table black_db.border ( PortName string, State string,
PortCode int, Border string, Date_ string, Value int,
Location_ string ) ROW FORMAT SERDE
'org.apache.hadoop.hive.serde2.OpenCSVSerde'
STORED AS INPUTFORMAT
'org.apache.hadoop.mapred.TextInputFormat' OUTPUTFORMAT
'org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat'
TBLPROPERTIES ( 'serialization.null.format' = '',
'skip.header.line.count' = '1')
```

LOAD DATA INPATH '/user/student4_2/Border_Crossing_Entry_Data.csv' INTO TABLE black_db.border ;

black_db

Tables (3)

Filter...

- appstore
- border
 - portname (string)
 - state (string)
 - portcode (string)
 - border (string)
 - date_ (string)
 - value (string)
 - location_ (string)
- forest

```
74 Date_       string,
75 Value       int,
76 Location_   string )
77 ROW FORMAT SERDE
78 'org.apache.hadoop.hive.serde2.OpenCSVSerde'
79 STORED AS INPUTFORMAT
80 'org.apache.hadoop.mapred.TextInputFormat'
81 OUTPUTFORMAT
82 'org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat'
83 TBLPROPERTIES (
84 'serialization.null.format' = '',
85 'skip.header.line.count' = '1');
86
87 LOAD DATA INPATH '/user/student4_2/Border_Crossing_Entry_Data.csv' INTO TABLE black_db.border;
88
```

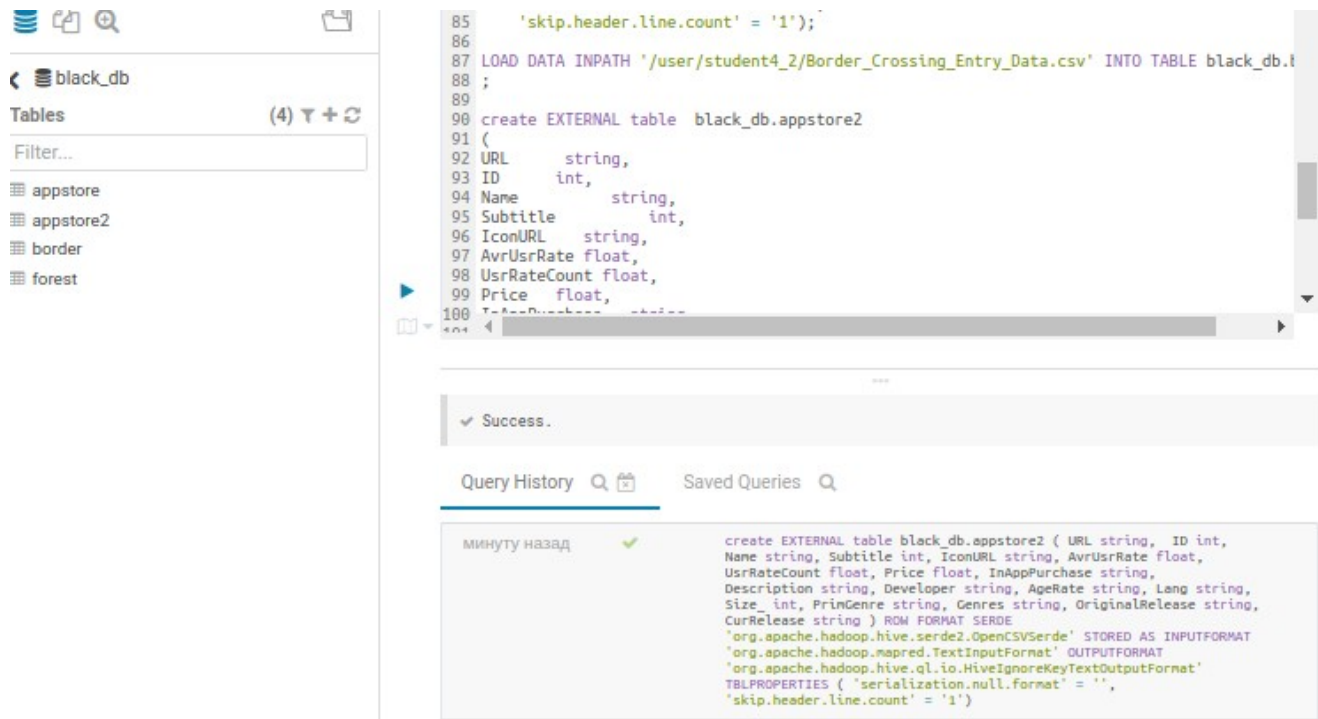
✓ Success.

Query History Saved Queries

несколько секунд назад

LOAD DATA INPATH
'/user/student4_2/Border_Crossing_Entry_Data.csv' INTO TABLE
black_db.border

Потом заметил что нужно делать external таблицы и сделал тоже самое с ключевым словом EXTERNAL.

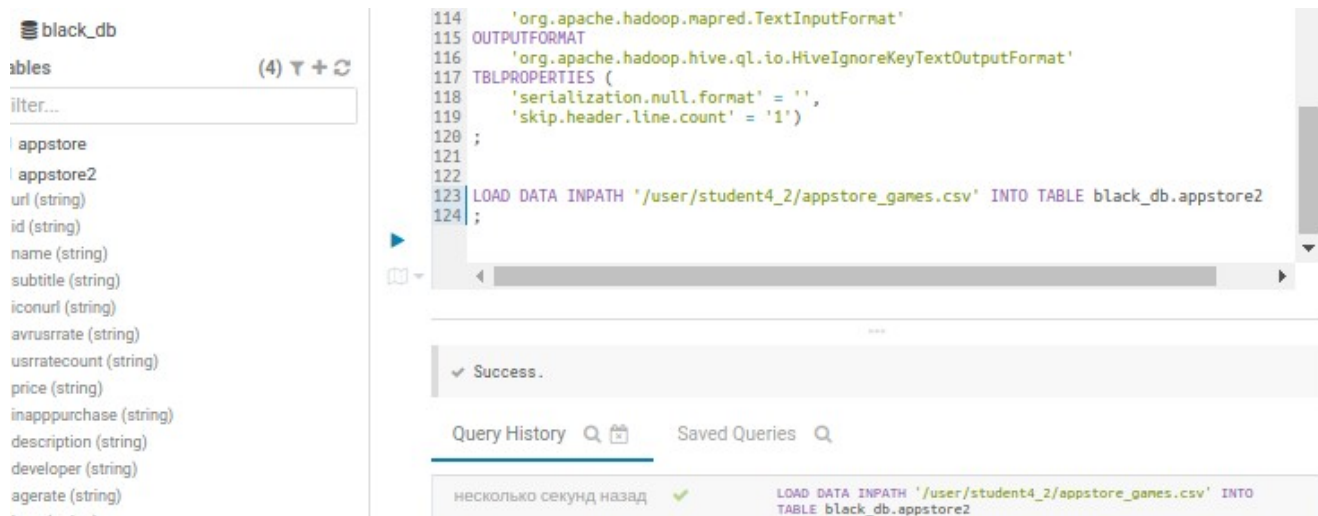


The screenshot shows a Hive CLI interface. On the left, a sidebar displays the database 'black_db' and a list of tables: 'appstore', 'appstore2', 'border', and 'forest'. The main area shows a SQL query being executed:

```
85 'skip.header.line.count' = '1');`
86
87 LOAD DATA INPATH '/user/student4_2/Border_Crossing_Entry_Data.csv' INTO TABLE black_db.l
88 ;
89
90 create EXTERNAL table black_db.appstore2
91 (
92 URL string,
93 ID int,
94 Name string,
95 Subtitle int,
96 IconURL string,
97 AvrUsrRate float,
98 UsrRateCount float,
99 Price float,
100 ...
101 ...
102 ...
103 ...
104 ...
105 ...
106 ...
107 ...
108 ...
109 ...
110 ...
111 ...
112 ...
113 ...
114 ...
115 ...
116 ...
117 ...
118 ...
119 ...
120 ...
121 ...
122 ...
123 ...
124 ...
125 ...
126 ...
127 ...
128 ...
129 ...
130 ...
131 ...
132 ...
133 ...
134 ...
135 ...
136 ...
137 ...
138 ...
139 ...
140 ...
141 ...
142 ...
143 ...
144 ...
145 ...
146 ...
147 ...
148 ...
149 ...
150 ...
151 ...
152 ...
153 ...
154 ...
155 ...
156 ...
157 ...
158 ...
159 ...
160 ...
161 ...
162 ...
163 ...
164 ...
165 ...
166 ...
167 ...
168 ...
169 ...
170 ...
171 ...
172 ...
173 ...
174 ...
175 ...
176 ...
177 ...
178 ...
179 ...
180 ...
181 ...
182 ...
183 ...
184 ...
185 ...
186 ...
187 ...
188 ...
189 ...
190 ...
191 ...
192 ...
193 ...
194 ...
195 ...
196 ...
197 ...
198 ...
199 ...
200 ...
201 ...
202 ...
203 ...
204 ...
205 ...
206 ...
207 ...
208 ...
209 ...
210 ...
211 ...
212 ...
213 ...
214 ...
215 ...
216 ...
217 ...
218 ...
219 ...
220 ...
221 ...
222 ...
223 ...
224 ...
225 ...
226 ...
227 ...
228 ...
229 ...
230 ...
231 ...
232 ...
233 ...
234 ...
235 ...
236 ...
237 ...
238 ...
239 ...
240 ...
241 ...
242 ...
243 ...
244 ...
245 ...
246 ...
247 ...
248 ...
249 ...
250 ...
251 ...
252 ...
253 ...
254 ...
255 ...
256 ...
257 ...
258 ...
259 ...
260 ...
261 ...
262 ...
263 ...
264 ...
265 ...
266 ...
267 ...
268 ...
269 ...
270 ...
271 ...
272 ...
273 ...
274 ...
275 ...
276 ...
277 ...
278 ...
279 ...
280 ...
281 ...
282 ...
283 ...
284 ...
285 ...
286 ...
287 ...
288 ...
289 ...
290 ...
291 ...
292 ...
293 ...
294 ...
295 ...
296 ...
297 ...
298 ...
299 ...
300 ...
301 ...
302 ...
303 ...
304 ...
305 ...
306 ...
307 ...
308 ...
309 ...
310 ...
311 ...
312 ...
313 ...
314 ...
315 ...
316 ...
317 ...
318 ...
319 ...
320 ...
321 ...
322 ...
323 ...
324 ...
325 ...
326 ...
327 ...
328 ...
329 ...
330 ...
331 ...
332 ...
333 ...
334 ...
335 ...
336 ...
337 ...
338 ...
339 ...
340 ...
341 ...
342 ...
343 ...
344 ...
345 ...
346 ...
347 ...
348 ...
349 ...
350 ...
351 ...
352 ...
353 ...
354 ...
355 ...
356 ...
357 ...
358 ...
359 ...
360 ...
361 ...
362 ...
363 ...
364 ...
365 ...
366 ...
367 ...
368 ...
369 ...
370 ...
371 ...
372 ...
373 ...
374 ...
375 ...
376 ...
377 ...
378 ...
379 ...
380 ...
381 ...
382 ...
383 ...
384 ...
385 ...
386 ...
387 ...
388 ...
389 ...
390 ...
391 ...
392 ...
393 ...
394 ...
395 ...
396 ...
397 ...
398 ...
399 ...
400 ...
401 ...
402 ...
403 ...
404 ...
405 ...
406 ...
407 ...
408 ...
409 ...
410 ...
411 ...
412 ...
413 ...
414 ...
415 ...
416 ...
417 ...
418 ...
419 ...
420 ...
421 ...
422 ...
423 ...
424 ...
425 ...
426 ...
427 ...
428 ...
429 ...
430 ...
431 ...
432 ...
433 ...
434 ...
435 ...
436 ...
437 ...
438 ...
439 ...
440 ...
441 ...
442 ...
443 ...
444 ...
445 ...
446 ...
447 ...
448 ...
449 ...
450 ...
451 ...
452 ...
453 ...
454 ...
455 ...
456 ...
457 ...
458 ...
459 ...
460 ...
461 ...
462 ...
463 ...
464 ...
465 ...
466 ...
467 ...
468 ...
469 ...
470 ...
471 ...
472 ...
473 ...
474 ...
475 ...
476 ...
477 ...
478 ...
479 ...
480 ...
481 ...
482 ...
483 ...
484 ...
485 ...
486 ...
487 ...
488 ...
489 ...
490 ...
491 ...
492 ...
493 ...
494 ...
495 ...
496 ...
497 ...
498 ...
499 ...
500 ...
501 ...
502 ...
503 ...
504 ...
505 ...
506 ...
507 ...
508 ...
509 ...
510 ...
511 ...
512 ...
513 ...
514 ...
515 ...
516 ...
517 ...
518 ...
519 ...
520 ...
521 ...
522 ...
523 ...
524 ...
525 ...
526 ...
527 ...
528 ...
529 ...
530 ...
531 ...
532 ...
533 ...
534 ...
535 ...
536 ...
537 ...
538 ...
539 ...
540 ...
541 ...
542 ...
543 ...
544 ...
545 ...
546 ...
547 ...
548 ...
549 ...
550 ...
551 ...
552 ...
553 ...
554 ...
555 ...
556 ...
557 ...
558 ...
559 ...
560 ...
561 ...
562 ...
563 ...
564 ...
565 ...
566 ...
567 ...
568 ...
569 ...
570 ...
571 ...
572 ...
573 ...
574 ...
575 ...
576 ...
577 ...
578 ...
579 ...
580 ...
581 ...
582 ...
583 ...
584 ...
585 ...
586 ...
587 ...
588 ...
589 ...
590 ...
591 ...
592 ...
593 ...
594 ...
595 ...
596 ...
597 ...
598 ...
599 ...
600 ...
601 ...
602 ...
603 ...
604 ...
605 ...
606 ...
607 ...
608 ...
609 ...
610 ...
611 ...
612 ...
613 ...
614 ...
615 ...
616 ...
617 ...
618 ...
619 ...
620 ...
621 ...
622 ...
623 ...
624 ...
625 ...
626 ...
627 ...
628 ...
629 ...
630 ...
631 ...
632 ...
633 ...
634 ...
635 ...
636 ...
637 ...
638 ...
639 ...
640 ...
641 ...
642 ...
643 ...
644 ...
645 ...
646 ...
647 ...
648 ...
649 ...
650 ...
651 ...
652 ...
653 ...
654 ...
655 ...
656 ...
657 ...
658 ...
659 ...
660 ...
661 ...
662 ...
663 ...
664 ...
665 ...
666 ...
667 ...
668 ...
669 ...
670 ...
671 ...
672 ...
673 ...
674 ...
675 ...
676 ...
677 ...
678 ...
679 ...
680 ...
681 ...
682 ...
683 ...
684 ...
685 ...
686 ...
687 ...
688 ...
689 ...
690 ...
691 ...
692 ...
693 ...
694 ...
695 ...
696 ...
697 ...
698 ...
699 ...
700 ...
701 ...
702 ...
703 ...
704 ...
705 ...
706 ...
707 ...
708 ...
709 ...
710 ...
711 ...
712 ...
713 ...
714 ...
715 ...
716 ...
717 ...
718 ...
719 ...
720 ...
721 ...
722 ...
723 ...
724 ...
725 ...
726 ...
727 ...
728 ...
729 ...
730 ...
731 ...
732 ...
733 ...
734 ...
735 ...
736 ...
737 ...
738 ...
739 ...
740 ...
741 ...
742 ...
743 ...
744 ...
745 ...
746 ...
747 ...
748 ...
749 ...
750 ...
751 ...
752 ...
753 ...
754 ...
755 ...
756 ...
757 ...
758 ...
759 ...
760 ...
761 ...
762 ...
763 ...
764 ...
765 ...
766 ...
767 ...
768 ...
769 ...
770 ...
771 ...
772 ...
773 ...
774 ...
775 ...
776 ...
777 ...
778 ...
779 ...
780 ...
781 ...
782 ...
783 ...
784 ...
785 ...
786 ...
787 ...
788 ...
789 ...
790 ...
791 ...
792 ...
793 ...
794 ...
795 ...
796 ...
797 ...
798 ...
799 ...
800 ...
801 ...
802 ...
803 ...
804 ...
805 ...
806 ...
807 ...
808 ...
809 ...
810 ...
811 ...
812 ...
813 ...
814 ...
815 ...
816 ...
817 ...
818 ...
819 ...
820 ...
821 ...
822 ...
823 ...
824 ...
825 ...
826 ...
827 ...
828 ...
829 ...
830 ...
831 ...
832 ...
833 ...
834 ...
835 ...
836 ...
837 ...
838 ...
839 ...
840 ...
841 ...
842 ...
843 ...
844 ...
845 ...
846 ...
847 ...
848 ...
849 ...
850 ...
851 ...
852 ...
853 ...
854 ...
855 ...
856 ...
857 ...
858 ...
859 ...
860 ...
861 ...
862 ...
863 ...
864 ...
865 ...
866 ...
867 ...
868 ...
869 ...
870 ...
871 ...
872 ...
873 ...
874 ...
875 ...
876 ...
877 ...
878 ...
879 ...
880 ...
881 ...
882 ...
883 ...
884 ...
885 ...
886 ...
887 ...
888 ...
889 ...
890 ...
891 ...
892 ...
893 ...
894 ...
895 ...
896 ...
897 ...
898 ...
899 ...
900 ...
901 ...
902 ...
903 ...
904 ...
905 ...
906 ...
907 ...
908 ...
909 ...
910 ...
911 ...
912 ...
913 ...
914 ...
915 ...
916 ...
917 ...
918 ...
919 ...
920 ...
921 ...
922 ...
923 ...
924 ...
925 ...
926 ...
927 ...
928 ...
929 ...
930 ...
931 ...
932 ...
933 ...
934 ...
935 ...
936 ...
937 ...
938 ...
939 ...
940 ...
941 ...
942 ...
943 ...
944 ...
945 ...
946 ...
947 ...
948 ...
949 ...
950 ...
951 ...
952 ...
953 ...
954 ...
955 ...
956 ...
957 ...
958 ...
959 ...
960 ...
961 ...
962 ...
963 ...
964 ...
965 ...
966 ...
967 ...
968 ...
969 ...
970 ...
971 ...
972 ...
973 ...
974 ...
975 ...
976 ...
977 ...
978 ...
979 ...
980 ...
981 ...
982 ...
983 ...
984 ...
985 ...
986 ...
987 ...
988 ...
989 ...
990 ...
991 ...
992 ...
993 ...
994 ...
995 ...
996 ...
997 ...
998 ...
999 ...
1000 ...
```

Below the query, a success message is displayed: "Success."

The Query History section shows the executed query with a green checkmark and the timestamp "минуту назад".



The screenshot shows a Hive CLI interface. On the left, a sidebar displays the database 'black_db' and a list of tables: 'appstore', 'appstore2', 'border', and 'forest'. The main area shows a SQL query being executed:

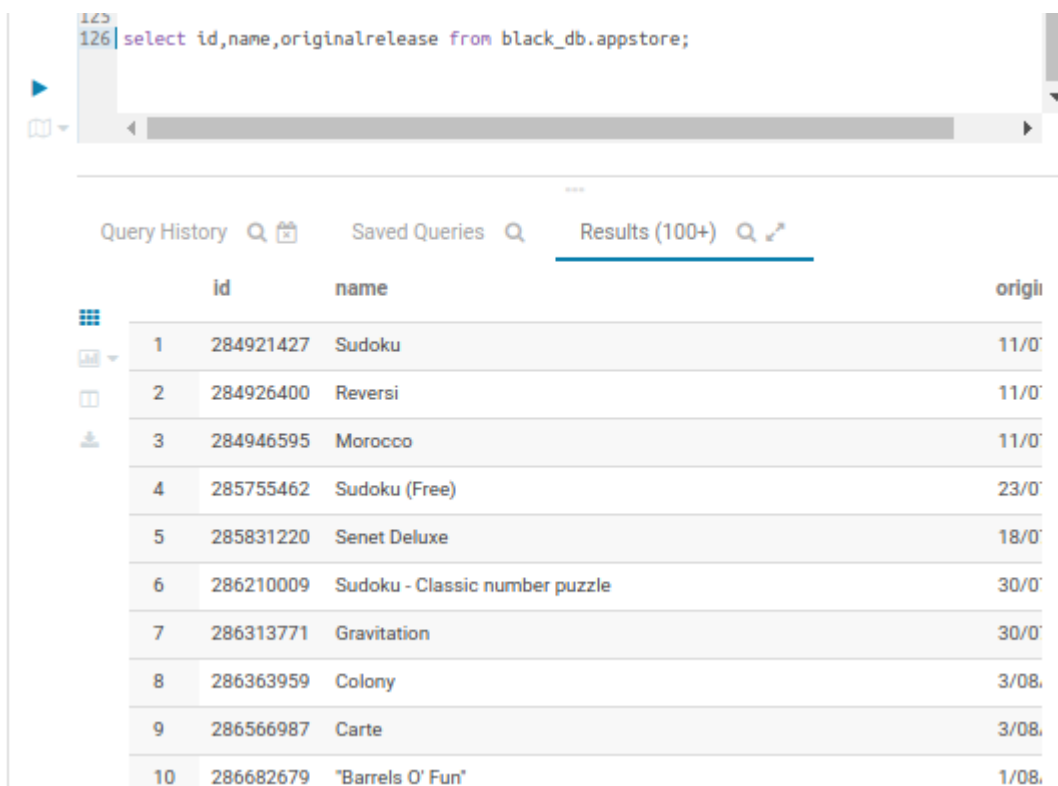
```
114 'org.apache.hadoop.mapred.TextInputFormat'
115 OUTPUTFORMAT
116 'org.apache.hadoop.hive ql.io.HiveIgnoreKeyTextOutputFormat'
117 TBLPROPERTIES (
118 'serialization.null.format' = '',
119 'skip.header.line.count' = '1')
120 ;
121
122
123 LOAD DATA INPATH '/user/student4_2/appstore_games.csv' INTO TABLE black_db.appstore2
124 ;
```

Below the query, a success message is displayed: "Success."

The Query History section shows the executed query with a green checkmark and the timestamp "несколько секунд назад".

Примеры запросов:

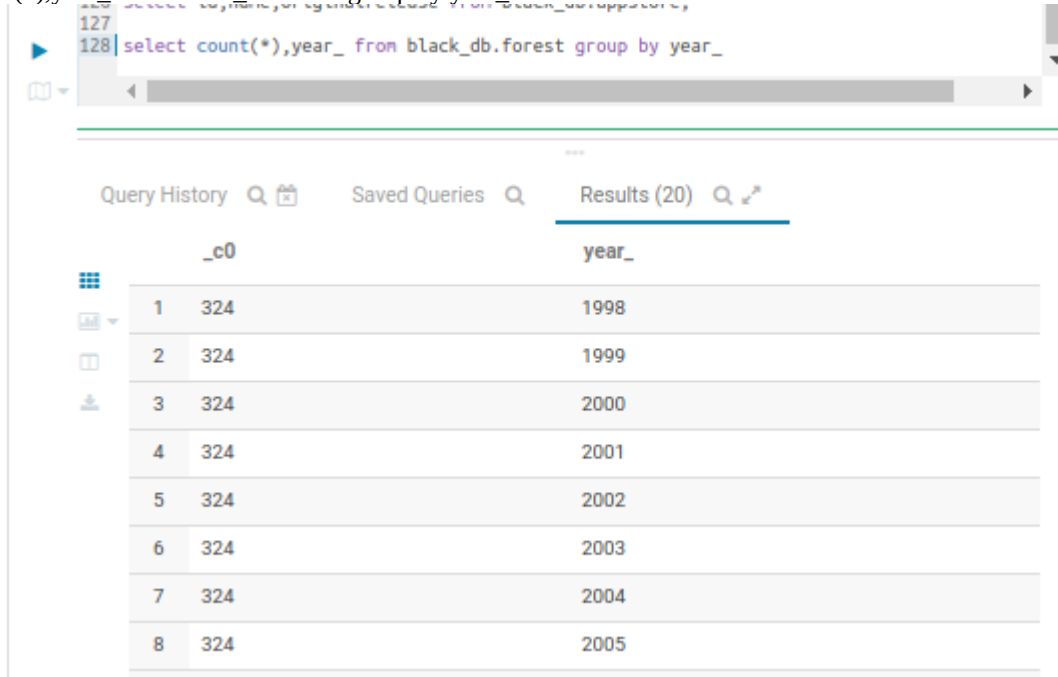
select id,name,originalrelease from black_db.appstore;



The screenshot shows a SQL query editor with a query entered in the top bar. Below the query bar, there are tabs for 'Query History', 'Saved Queries', and 'Results (100+)'. The 'Results (100+)' tab is active, displaying a table with 10 rows of data. The table has columns for 'id', 'name', and 'originalrelease'.

	id	name	originalrelease
1	284921427	Sudoku	11/0'
2	284926400	Reversi	11/0'
3	284946595	Morocco	11/0'
4	285755462	Sudoku (Free)	23/0'
5	285831220	Senet Deluxe	18/0'
6	286210009	Sudoku - Classic number puzzle	30/0'
7	286313771	Gravitation	30/0'
8	286363959	Colony	3/08,
9	286566987	Carte	3/08,
10	286682679	"Barrels O' Fun"	1/08,

select count(*),year_ from black_db.forest group by year_



The screenshot shows a SQL query editor with a query entered in the top bar. Below the query bar, there are tabs for 'Query History', 'Saved Queries', and 'Results (20)'. The 'Results (20)' tab is active, displaying a table with 8 rows of data. The table has columns for '_c0' and 'year_'.

	c0	year
1	324	1998
2	324	1999
3	324	2000
4	324	2001
5	324	2002
6	324	2003
7	324	2004
8	324	2005