Class 6 BigQuery

Elements of Databases

Mar 4, 2022

Announcements

- GCP credit check (Instapoll)
- How to request additional GCP credits:
  - Follow this guide
  - Only one person per group should request credit

Midterm 1

- When: Next class (03/11 at 4pm)
- Where: Home
- Duration: 90 minutes
- How: Canvas Quiz
- Format:
  - T/F section (10-12 questions)
  - MC section (10-12 questions)
  - Coding section (4-5 questions)
- Review session: Tues 03/08 from 11:30am - 1pm
- Practice Exam: Will be shared before review session

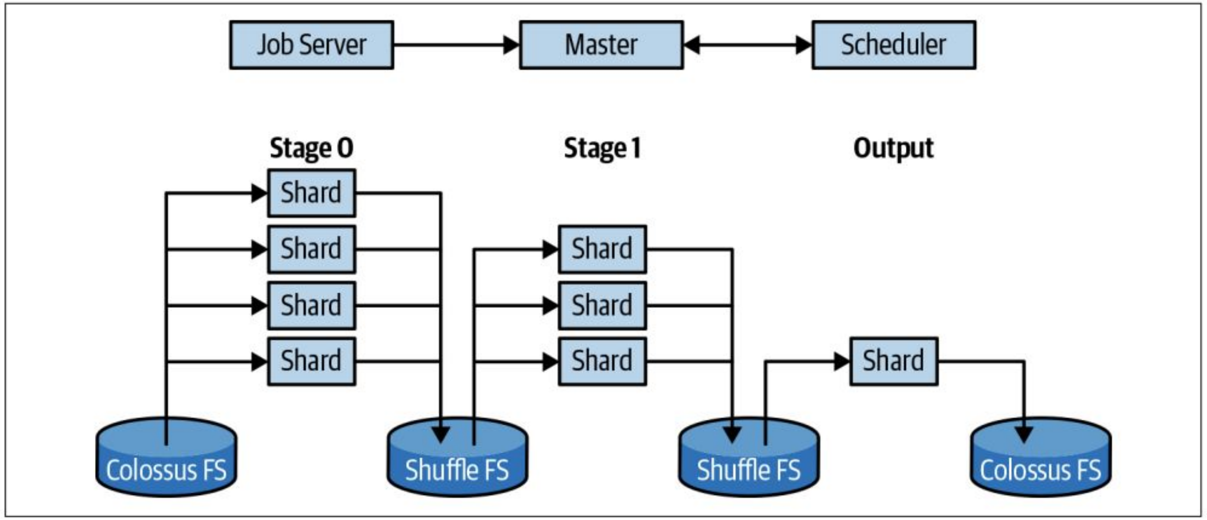
Exam Rules:

- Open-notes
- Open-book
- Open-project
- Do not crowdsource your notes
- Do not consult with any humans in any form during exam
- Piazza will be disabled for new posts, enabled for old posts

BigQuery’s Architecture

Key design principle:
Storage and compute are scaled independently.

Example query:
SELECT a, b, c, COUNT(*)
FROM T
GROUP BY a, b, c
ORDER BY a;



Source: Google BigQuery: The Definitive Guide (2019).

Views

- Return a table of results from a SQL query
- Saved in the database as named query
- Defined by `CREATE VIEW` statement

Employee(<u>empid</u>, fname, lname, job_function, level, title, manager_id, start_date,
salary, dob, ssn, emergency_contact)

```
CREATE VIEW Direct_Manager_Org AS
    SELECT empid, fname, lname, job_function, level, title,
               manager_id, start_date, salary, dob
    FROM Employee
    WHERE manager_id = 'abc'
    ORDER BY empid;
```

```
SELECT empid, fname, lname
FROM Direct_Manager_Org
WHERE start_date < '2020-01-01'
AND title = 'Data Engineer'
```

What’s wrong with these queries?

Employee(<u>empid</u>, fname, lname, job_function, level, title, manager_id, start_date,
salary, dob, ssn, emergency_contact)

```
CREATE VIEW Director_Org AS
   SELECT empid, fname, lname, job_function, level
   FROM Employee
   WHERE level NOT IN ('SVP', 'VP', 'CEO')
   ORDER BY empid;

  SELECT empid, fname, lname
  FROM Director_Org
  WHERE salary > 300000
  AND level = 'Director';
```

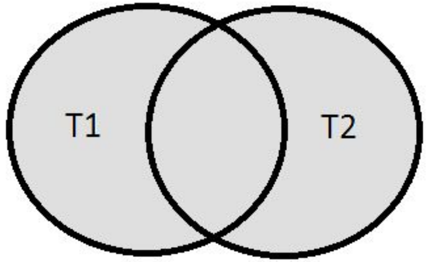
```
CREATE VIEW Senior_Manager_Org AS
   SELECT empid, fname, lname, job_function, level,
            start_date, salary
   FROM Director_Org
   WHERE level != 'Director'
   AND manager_id = 123
   ORDER BY empid;

  SELECT empid, fname, lname
  FROM Senior_Manager_Org
  WHERE start_date < '2020-01-01'
  AND job_function = 'ENG';
```

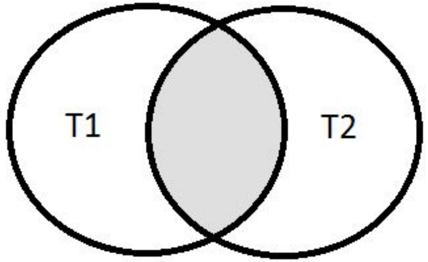
Optional Readings: Chapter 6 from Learning SQL.

Set Operations

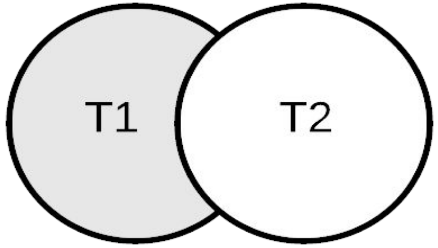
```
SELECT a, b, c FROM T1
UNION ALL | DISTINCT
SELECT a, b, c FROM T2;
```



```
SELECT a, b, c FROM T1
INTERSECT DISTINCT
SELECT a, b, c FROM T2;
```



```
SELECT a, b, c FROM T1
EXCEPT DISTINCT
SELECT a, b, c FROM T2;
```



Subqueries

```
SELECT a, b, c
FROM T1
WHERE a =
      (SELECT x FROM T2 ...)
```

Comparison Operators:

=

!=

>

<

<=

>=

- Subqueries can be attached to nearly every clause of a query
- Two major types of subqueries: uncorrelated and correlated
- Parenthesis around subquery required

Subqueries in the `WHERE` clause

```
SELECT a, b, c
FROM T1
WHERE d IN
      (SELECT x FROM T2 ...)
```

List Membership Operators:

`IN`

`NOT IN`

Comparison Operators:

`=, !=, >, <, <=, >=`

Exercise 1: Subqueries

Who are the oldest students?

Database Schema:

Student(<u>sid</u>, fname, lname, dob, status)

Class(<u>cno</u>, cname, credits)

Instructor(<u>tid</u>, fname, lname, dept)

Takes(<u>sid</u>, <u>cno</u>, grade)

Teaches(<u>tid</u>, <u>cno</u>)

Exercise 2: Set Operation

Who takes CS327E ***and*** CS331E?

Return the sid, first and last names of the students who take both classes.

Order the results by last name, followed by first name.

Database Schema:

Student(<u>sid</u>, fname, lname, dob, status)

Class(<u>cno</u>, cname, credits)

Instructor(<u>tid</u>, fname, lname, dept)

Takes(<u>sid</u>, <u>cno</u>, grade)

Teaches(<u>tid</u>, <u>cno</u>)

Exercise 3: Subqueries

*Who does **not** take CS327E?*

Return the sid, first and last names of the students who don't take the class.

Order the results by last name, followed by first name.

Database Schema:

Student(<u>sid</u>, fname, lname, dob, status)

Class(<u>cno</u>, cname, credits)

Instructor(<u>tid</u>, fname, lname, dept)

Takes(<u>sid</u>, <u>cno</u>, grade)

Teaches(<u>tid</u>, <u>cno</u>)

Subqueries in the `FROM` and `JOIN` clauses

```
SELECT a, b, c
FROM (SELECT a, b, c FROM U ...)
[WHERE ...]
[ORDER BY ...]

SELECT a, b, c, d, e, f
FROM (SELECT a, b, c FROM U ...) JOIN T
ON a = d
[WHERE ... ORDER BY ...]
```

Subqueries in HAVING clause

```
SELECT a, b, c <aggregate functions>
FROM T1
[WHERE <boolean condition>]
GROUP BY a, b, c
HAVING <aggregate function> = (SELECT x
                               FROM T2 ...)
```

Comparison Operators: **=, !=, >, <, <=, >=**

Exercise 4: Subqueries

Which classes have a higher enrollment than the overall average enrollment per class?

Return the cno and the enrollment count for those classes.

Database Schema:

Student(<u>sid</u>, fname, lname, dob, status)

Class(<u>cno</u>, cname, credits)

Instructor(<u>tid</u>, fname, lname, dept)

Takes(<u>sid</u>, <u>cno</u>, grade)

Teaches(<u>tid</u>, <u>cno</u>)

No need to account for classes with zero enrollment.

Correlated Subqueries in the `WHERE` clause

```
SELECT a, b, c
FROM T
WHERE c > (SELECT d FROM U WHERE U.e = T.b)
```

Comparison Operators: `=, !=, >, <, <=, >=`

List Membership Operators: `IN, NOT IN`

Subqueries in the `SELECT` clause

```
SELECT a, b, c, (SELECT aggr. FROM U [WHERE U.e = T.b])
FROM T
[WHERE ... ]
```

Example:

```
select distinct sid,
    (select min(grade) from
     college.Takes u
     where u.sid = t.sid)
from college.Takes t;
```

Database Schema:

Student(sid, fname, lname, dob,
status)

Class(cno, cname, credits)

Instructor(tid, fname, lname, dept)

Takes(sid, cno, grade)

Teaches(tid, cno)

Exercise 5: Subqueries

Which instructors earn a higher salary than the average salary of their department?

Return the instructor's name, department, and salary.

Order the results by salary in descending order.

Database Schema:

Student(sid, fname, lname, dob, status)

Class(cno, cname, credits)

Instructor(tid, name, dept, **sal**)

Takes(sid, cno, grade)

Teaches(tid, cno)