SQL commands:

Original (Q1):

```
-- Q1, Find conferences held in 2018 that have published at least 200 papers in a single decade
WITH win1 AS (
   SELECT booktitle
   FROM inproceedings
   WHERE year BETWEEN '2008' AND '2018'
   GROUP BY booktitle
   HAVING COUNT(*) >= 200
)
SELECT DISTINCT booktitle
FROM inproceedings
WHERE year = '2018'
  AND booktitle IN (SELECT booktitle FROM win1);
```

EXPLAIN (Original):

| | QUERY PLAN<br>text |
|---|---|
| 1 | HashAggregate (cost=232518.43..232577.65 rows=5922 width=8) |
| 2 | Group Key: inproceedings.booktitle |
| 3 | -> Hash Join (cost=112890.42..232375.59 rows=57136 width=8) |
| 4 | Hash Cond: (inproceedings.booktitle = inproceedings_1.booktitle) |
| 5 | -> Gather (cost=1000.00..120034.90 rows=171407 width=8) |
| 6 | Workers Planned: 2 |
| 7 | -> Parallel Seq Scan on inproceedings (cost=0.00..101894.20 rows=71420 width=8) |
| 8 | Filter: (year = '2018'::text) |
| 9 | -> Hash (cost=111865.75..111865.75 rows=1974 width=8) |
| 10 | -> Finalize GroupAggregate (cost=110350.61..111865.75 rows=1974 width=8) |
| 11 | Group Key: inproceedings_1.booktitle |
| 12 | Filter: (count(*) >= 200) |
| 13 | -> Gather Merge (cost=110350.61..111732.50 rows=11844 width=16) |
| 14 | Workers Planned: 2 |
| 15 | -> Sort (cost=109350.58..109365.39 rows=5922 width=16) |
| 16 | Sort Key: inproceedings_1.booktitle |
| 17 | -> Partial HashAggregate (cost=108920.29..108979.51 rows=5922 width=16) |
| 18 | Group Key: inproceedings_1.booktitle |
| 19 | -> Parallel Seq Scan on inproceedings inproceedings_1 (cost=0.00..105530.64 rows=677930 widt... |
| 20 | Filter: ((year >= '2008'::text) AND (year <= '2018'::text)) |

Runtime with EXPLAIN ANALYZE (Original):

| | |
|---|---|
| 34 | Planning Time: 0.281 ms |
| 35 | Execution Time: 2292.075 ms |

Runtime after Indexing with
```
CREATE INDEX idx_inproceedings_year ON inproceedings (year);
CREATE INDEX idx_inproceedings_booktitle ON inproceedings (booktitle);
```

| |
|---|
| Planning Time: 1.697 ms |
| Execution Time: 2283.611 ms |

Changing the Query – Use join instead of in, and change the window selection to avoid repeats
```
WITH win1 AS (
  SELECT booktitle
  FROM inproceedings
  WHERE year BETWEEN '2008' AND '2017'
  GROUP BY booktitle
  HAVING COUNT(*) >= 200
)
SELECT DISTINCT i.booktitle
FROM inproceedings i
JOIN win1 w ON i.booktitle = w.booktitle
WHERE i.year = '2018';
```
Runtime after all changes:

| |
|---|
| Planning Time: 0.319 ms |
| Execution Time: 2204.469 ms |

##
Original (Q2):
```
-- Q2, Find Authors who published at least 10 PVLDB papers and at least 10 SIGMOD papers
WITH win1 AS (
  SELECT p.author
  FROM inproceedings p
  JOIN articles a ON a.author = p.author
  WHERE (p.booktitle LIKE '%VLDB%' OR a.title LIKE '%VLDB%')
    AND p.author IS NOT NULL
  GROUP BY p.author
```

```
  HAVING COUNT(*) >= 10
),
win2 AS (
  SELECT p.author
  FROM inproceedings p
  JOIN articles a ON a.author = p.author
  WHERE (p.booktitle LIKE '%SIGMOD%' OR a.title LIKE '%SIGMOD%')
    AND p.author IS NOT NULL
  GROUP BY p.author
  HAVING COUNT(*) >= 10
)
SELECT a.author
FROM win1 a
JOIN win2 b ON a.author = b.author;
```

EXPLAIN (Original):

| | QUERY PLAN<br>text |
|---|---|
| 1 | Hash Join (cost=668592.29..668733.19 rows=753 width=54) |
| 2 | Hash Cond: (p.author = p_1.author) |
| 3 | -> Finalize GroupAggregate (cost=334223.79..334363.65 rows=388 width=54) |
| 4 | Group Key: p.author |
| 5 | Filter: (count(*) >= 10) |
| 6 | -> Gather Merge (cost=334223.79..334344.24 rows=970 width=62) |
| 7 | Workers Planned: 2 |
| 8 | -> Partial GroupAggregate (cost=333223.76..333232.25 rows=485 width=62) |
| 9 | Group Key: p.author |
| 10 | -> Sort (cost=333223.76..333224.98 rows=485 width=54) |
| 11 | Sort Key: p.author |
| 12 | -> Parallel Hash Join (cost=132007.45..333202.13 rows=485 width=54) |
| 13 | Hash Cond: (a.author = p.author) |
| 14 | Join Filter: ((p.booktitle ~~ '%VLDB%'::text) OR (a.title ~~ '%VLDB%'::text)) |
| 15 | -> Parallel Seq Scan on articles a (cost=0.00..105071.05 rows=1470305 width=133) |
| 16 | -> Parallel Hash (cost=98257.76..98257.76 rows=1452055 width=62) |
| 17 | -> Parallel Seq Scan on inproceedings p (cost=0.00..98257.76 rows=1452055 width=62) |
| 18 | Filter: (author IS NOT NULL) |
| 19 | -> Hash (cost=334363.65..334363.65 rows=388 width=54) |
| 20 | -> Finalize GroupAggregate (cost=334223.79..334363.65 rows=388 width=54) |
| 21 | Group Key: p_1.author |
| 22 | Filter: (count(*) >= 10) |
| 23 | -> Gather Merge (cost=334223.79..334344.24 rows=970 width=62) |
| 24 | Workers Planned: 2 |
| 25 | -> Partial GroupAggregate (cost=333223.76..333232.25 rows=485 width=62) |
| 26 | Group Key: p_1.author |
| 27 | -> Sort (cost=333223.76..333224.98 rows=485 width=54) |
| 28 | Sort Key: p_1.author |
| 29 | -> Parallel Hash Join (cost=132007.45..333202.13 rows=485 width=54) |
| 30 | Hash Cond: (a_1.author = p_1.author) |
| 31 | Join Filter: ((p_1.booktitle ~~ '%SIGMOD%'::text) OR (a_1.title ~~ '%SIGMOD%'::text)) |
| 32 | -> Parallel Seq Scan on articles a_1 (cost=0.00..105071.05 rows=1470305 width=133) |
| 33 | -> Parallel Hash (cost=98257.76..98257.76 rows=1452055 width=62) |
| 34 | -> Parallel Seq Scan on inproceedings p_1 (cost=0.00..98257.76 rows=1452055 width... |
| 35 | Filter: (author IS NOT NULL) |

EXPLAIN ANALYZE (Original):

| | |
|---|---|
| 53 | Planning Time: 0.717 ms |
| 54 | Execution Time: 32423.811 ms |

Runtime after Indexing with
```
CREATE INDEX idx_inproceedings_author ON inproceedings (LEFT(author, 255));
CREATE INDEX idx_articles_author ON articles (LEFT(author, 255));
```

| | |
|---|---|
| 53 | Planning Time: 1.000 ms |
| 54 | Execution Time: 23227.148 ms |

Original (Q3):
```
-- Q3, Find Total Conference Publications for each decade, starting from 1970 and ending in 2019
with win1 AS(
        SELECT year, count(*)
        FROM publications
        WHERE year is NOT NULL
        GROUP BY year
)
SELECT
  SUM(CASE WHEN year BETWEEN '1970' AND '1979' THEN count ELSE 0 END) AS "1970s",
  SUM(CASE WHEN year BETWEEN '1980' AND '1989' THEN count ELSE 0 END) AS "1980s",
        SUM(CASE WHEN year BETWEEN '1990' AND '1999' THEN count ELSE 0 END) AS "1990s",
        SUM(CASE WHEN year BETWEEN '2000' AND '2009' THEN count ELSE 0 END) AS "2000s",
        SUM(CASE WHEN year BETWEEN '2010' AND '2019' THEN count ELSE 0 END) AS "2010s"
FROM win1;
```

Results from EXPLAIN (Original)

| | QUERY PLAN<br>text |
|---|---|
| 1 | Aggregate (cost=162005.37..162005.38 rows=1 width=160) |
| 2 | -> Finalize GroupAggregate (cost=161984.42..162002.66 rows=72 width=13) |
| 3 | Group Key: publications.year |
| 4 | -> Gather Merge (cost=161984.42..162001.22 rows=144 width=13) |
| 5 | Workers Planned: 2 |
| 6 | -> Sort (cost=160984.39..160984.57 rows=72 width=13) |
| 7 | Sort Key: publications.year |
| 8 | -> Partial HashAggregate (cost=160981.45..160982.17 rows=72 width=13) |
| 9 | Group Key: publications.year |
| 10 | -> Parallel Seq Scan on publications (cost=0.00..146168.97 rows=2962497 width... |
| 11 | Filter: (year IS NOT NULL) |

Results for EXPLAIN ANALYZE (Original)

| | QUERY PLAN<br>text |
|---|---|
| 1 | Aggregate (cost=162005.37..162005.38 rows=1 width=160) (actual time=13834.659..13838.525 rows=1 loops=1) |
| 2 | -> Finalize GroupAggregate (cost=161984.42..162002.66 rows=72 width=13) (actual time=13833.243..13837.231 rows=89 loops=1) |
| 3 | Group Key: publications.year |
| 4 | -> Gather Merge (cost=161984.42..162001.22 rows=144 width=13) (actual time=13833.236..13837.179 rows=252 loops=1) |
| 5 | Workers Planned: 2 |
| 6 | Workers Launched: 2 |
| 7 | -> Sort (cost=160984.39..160984.57 rows=72 width=13) (actual time=13800.433..13800.438 rows=84 loops=3) |
| 8 | Sort Key: publications.year |
| 9 | Sort Method: quicksort Memory: 28kB |
| 10 | Worker 0: Sort Method: quicksort Memory: 27kB |
| 11 | Worker 1: Sort Method: quicksort Memory: 28kB |
| 12 | -> Partial HashAggregate (cost=160981.45..160982.17 rows=72 width=13) (actual time=13800.239..13800.248 rows=84 loops=3) |
| 13 | Group Key: publications.year |
| 14 | Batches: 1 Memory Usage: 24kB |
| 15 | Worker 0: Batches: 1 Memory Usage: 24kB |
| 16 | Worker 1: Batches: 1 Memory Usage: 24kB |
| 17 | -> Parallel Seq Scan on publications (cost=0.00..146168.97 rows=2962497 width=5) (actual time=1.003..13220.150 rows=2370129 loo... |
| 18 | Filter: (year IS NOT NULL) |
| 19 | Rows Removed by Filter: 1 |
| 20 | Planning Time: 0.189 ms |
| 21 | Execution Time: 13838.605 ms |

Runtime after Indexing with

```

CREATE INDEX idx_publications_year ON publications (year);

```

| | |
|---|---|
| 12 | Planning Time: 1.466 ms |
| 13 | Execution Time: 948.252 ms |

##

Q4 (Original):
```
-- Q4, Find top 10 authors publishing in journals and conferences that have the word 'data' in the title
WITH win AS (
	SELECT a.author, a.title FROM articles a
	WHERE a.title LIKE '%Data%'
	AND a.author IS NOT NULL
	UNION
	SELECT p.author, p.title FROM inproceedings p
	WHERE p.title LIKE '%Data%'
	AND p.author IS NOT NULL
	)

SELECT author, COUNT(*) AS num_titles FROM win
GROUP BY author
ORDER BY num_titles DESC
LIMIT 10;
```

EXPLAIN:

| | QUERY PLAN<br>text |
|---|---|
| 1 | Limit  (cost=264595.03..264595.05 rows=10 width=40) |
| 2 | -> Sort  (cost=264595.03..264595.53 rows=200 width=40) |
| 3 | Sort Key: (count(*)) DESC |
| 4 | -> GroupAggregate  (cost=259835.49..264590.70 rows=200 width=40) |
| 5 | Group Key: a.author |
| 6 | -> Unique  (cost=259835.49..261419.89 rows=211254 width=64) |
| 7 | -> Sort  (cost=259835.49..260363.62 rows=211254 width=64) |
| 8 | Sort Key: a.author, a.title |
| 9 | -> Gather  (cost=1000.00..233206.53 rows=211254 width=64) |
| 10 | Workers Planned: 2 |
| 11 | -> Parallel Append  (cost=0.00..211081.13 rows=88023 width=64) |
| 12 | -> Parallel Seq Scan on articles a  (cost=0.00..108746.81 rows=14706 width=133) |
| 13 | Filter: ((author IS NOT NULL) AND (title ~~ '%Data%'::text)) |
| 14 | -> Parallel Seq Scan on inproceedings p  (cost=0.00..101894.20 rows=73317 width=... |
| 15 | Filter: ((author IS NOT NULL) AND (title ~~ '%Data%'::text)) |

EXPLAIN ANALYZE (Original):

| | |
|---|---|
| 21 | Planning Time: 0.325 ms |
| 22 | Execution Time: 5890.889 ms |

Runtime after Indexing with
```
CREATE INDEX idx_articles_author ON articles (author);
CREATE INDEX idx_inproceedings_author ON inproceedings (author);
```

| | |
|---|---|
| 21 | Planning Time: 0.413 ms |
| 22 | Execution Time: 5072.486 ms |

Q5 (Original):
```
-- Q5, find all conferences happening in June (Assume means recent, year 2018), where proceedings have more than 100 publications

WITH publications AS (
   SELECT booktitle, COUNT(*) AS number_of_publications, CAST(year AS INTEGER) AS year
```

```
  FROM INPROCEEDINGS
  GROUP BY booktitle, year
  UNION ALL
  SELECT booktitle, COUNT(*) AS number_of_publications, CAST(year AS INTEGER) AS year
  FROM PROCEEDINGS
  GROUP BY booktitle, year
)
SELECT booktitle
FROM publications
WHERE year = 2018 AND number_of_publications > 100;
```

EXPLAIN (Original):

| | QUERY PLAN<br>text |
|---|---|
| 1 | Append (cost=110633.61..115962.18 rows=5786 width=8) |
| 2 | -> Subquery Scan on "*SELECT* 1" (cost=110633.61..112837.36 rows=5689 width=8) |
| 3 | -> Finalize GroupAggregate (cost=110633.61..112780.47 rows=5689 width=25) |
| 4 | Group Key: inproceedings.booktitle, inproceedings.year |
| 5 | Filter: (count(*) > 100) |
| 6 | -> Gather Merge (cost=110633.61..112458.04 rows=14546 width=21) |
| 7 | Workers Planned: 2 |
| 8 | -> Partial GroupAggregate (cost=109633.59..109779.05 rows=7273 width=21) |
| 9 | Group Key: inproceedings.booktitle, inproceedings.year |
| 10 | -> Sort (cost=109633.59..109651.77 rows=7273 width=13) |
| 11 | Sort Key: inproceedings.booktitle, inproceedings.year |
| 12 | -> Parallel Seq Scan on inproceedings (cost=0.00..109167.08 rows=7273 width=... |
| 13 | Filter: ((year)::integer = 2018) |
| 14 | -> Subquery Scan on "*SELECT* 2" (cost=3088.34..3095.88 rows=97 width=10) |
| 15 | -> GroupAggregate (cost=3088.34..3094.91 rows=97 width=27) |
| 16 | Group Key: proceedings.booktitle, proceedings.year |
| 17 | Filter: (count(*) > 100) |
| 18 | -> Sort (cost=3088.34..3089.07 rows=294 width=15) |
| 19 | Sort Key: proceedings.booktitle, proceedings.year |
| 20 | -> Seq Scan on proceedings (cost=0.00..3076.28 rows=294 width=15) |
| 21 | Filter: ((year)::integer = 2018) |

EXPLAIN ANALYZE (Original)

| | |
|---|---|
| | Planning Time: 0.253 ms |
| | Execution Time: 836.347 ms |

After indexing with
```
CREATE INDEX idx_inproceedings_year_booktitle ON inproceedings (year, booktitle);
CREATE INDEX idx_proceedings_year_booktitle ON proceedings (year, booktitle);
```

| | |
|---|---|
| 32 | Planning Time: 0.622 ms |
| 33 | Execution Time: 910.771 ms |

Making some changes: Union first, and then do a single GROUPBY. CAST is expensive, do it once instead of twice. Changed query:
```
WITH publications AS (
  SELECT booktitle, COUNT(*) AS number_of_publications, CAST(year AS INTEGER) AS year
  FROM (
    SELECT booktitle, year FROM inproceedings
    UNION ALL
    SELECT booktitle, year FROM proceedings
  )
  GROUP BY booktitle, year
)
SELECT booktitle
FROM publications
WHERE year = 2018 AND number_of_publications > 100;
```
Results after all changes:

| | |
|---|---|
| 31 | Planning Time: 0.263 ms |
| 32 | Execution Time: 830.428 ms |

Query Optimizing

1. Queries can be optimized by either rewriting the query so that it is more efficient, such as avoiding anything that scans the entire table. Reorganizing the order of filters within the query and optimizing joins can also lead to faster runtimes.
2. Creating indexes for popular columns should also improve runtimes. Below is a list of indexes that were created. They are picked from common targets used by all 5 queries.
3. Usually after indexing, the query cost planning time increased but the execution time decreased.

##

Indexing targets combined for the 5 questions – Running this first will work on all 5 questions

```
CREATE INDEX idx_inproceedings_year_booktitle ON INPROCEEDINGS (LEFT(author, 255), year, booktitle);
CREATE INDEX idx_proceedings_year_booktitle ON PROCEEDINGS (year, booktitle);
CREATE INDEX idx_articles_author ON articles (LEFT(author, 255), title);
CREATE INDEX idx_publications_year ON publications (year);
```