Praktikum 5 Dokumentation

1. Ausgabe aller Spieler (Spielername), die in einem bestimmten Zeitraum gespielt hatten.

Native SQL (zu Testzwecken):

```
SELECT p.pid, p.name FROM Player p, Gamesession gs  WHERE
(((gs.timestampstart > '2019-06-10 00:00:00.0')  AND
(gs.timestampend < '2019-06-15 00:00:00.0')) AND (p.pid =
gs.player_pid));
```

Generierte SQL Query:

```
SELECT t0.NAME FROM PLAYER t0, GAMESESSION t1 WHERE
(((t1.TIMESTAMPSTART > ?) AND (t1.TIMESTAMPEND < ?)) AND (t0.PID =
t1.PLAYER_PID))
```

bind => [2019-04-28 00:00:00.0, 2019-04-29 00:00:00.0]

JPQL Query:

```
SELECT gs.player.name FROM Gamesession AS gs WHERE gs.timeStampStart >
:startdate AND gs.timeStampEnd < :enddate
```

Indizes:

CREATE INDEX index_timestampstart on public.gamesession USING BTREE (timestampstart)

CREATE INDEX index_timestampend on public.gamesession USING BTREE (timestampend)

	Vor der Optimierung	Nach der Optimierung
Planning Time:	0.601 ms	1.028 ms
Execution Time:	293.767 ms	42.726 ms

2. Ausgabe zu einem bestimmten Spieler: Alle Spiele (Id, Datum), sowie die Anzahl der korrekten Antworten pro Spiel mit Angabe der Gesamtanzahl der Fragen pro Spiel bzw. alternativ den Prozentsatz der korrekt beantworteten Fragen.

Native SQL (zu Testzwecken):

```
SELECT gs.gsid, gs.timestampstart, gs.timestampend,
gs.correctanswers, COUNT(q.qid)
FROM gamesession AS gs, Question AS q, Player AS pl,
gamesession_question AS gsq
WHERE ((pl.name = 'Albin') AND (pl.pid = gs.player_pid)
AND ((gsq.gamesession_gsid = gs.gsid) AND (q.qid =
gsq.askedquestions_qid))) GROUP BY gs.gsid;
```

Generierte SQL Query:

```
SELECT t0.GSID, t0.TIMESTAMPEND, t0.TIMESTAMPEND, t0.CORRECTANSWERS,
COUNT(t1.QID) FROM GAMESESSION_Question t3, PLAYER t2, Question t1,
GAMESESSION t0 WHERE ((t2.NAME = ?) AND ((t2.PID = t0.PLAYER_PID)
AND ((t3.Gamesession_GSID = t0.GSID) AND (t1.QID =
t3.askedQuestions_QID)))) GROUP BY t0.GSID
```

bind => [Albin]

JPQL Query:

```
SELECT gs.gsid, gs.timeStampEnd, gs.timeStampEnd, gs.correctAnswers
, COUNT(gs.askedQuestions) FROM Gamesession AS gs WHERE
gs.player.name = :playername GROUP BY gs.gsid
```

Index:

CREATE INDEX index_gs_player ON public.gamesession USING HASH (player_pid)

	Vor der Optimierung	Nach der Optimierung
Planning Time:	2.582 ms	3.107 ms
Execution Time:	337.795 ms	7.137 ms

3. Ausgabe aller Spieler mit Anzahl der gespielten Spiele, nach Anzahl absteigend geordnet.

Native SQL (zu Testzwecken):

```
SELECT p.name, COUNT(gs.gsid) FROM Player AS p, Gamesession AS gs
WHERE (p.pid = gs.player_pid) GROUP BY p.name ORDER BY
COUNT(gs.gsid) DESC
```

Generierte SQL Query:

```
SELECT t0.name, COUNT(t1.GSID) FROM GAMESESSION_CATEGORY t3,
CATEGORY t2, GAMESESSION t1, CATEGORY t0 WHERE ((t0.CID = t2.CID)
AND ((t3.Gamesession_GSID = t1.GSID) AND (t2.CID =
t3.selectedCategories_CID))) GROUP BY t0.name ORDER BY
COUNT(t1.GSID) DESC
```

JPQL Query:

```
SELECT gs.player.name, COUNT(gs.gsid) FROM Gamesession AS gs
GROUP BY gs.player.name ORDER BY COUNT(gs.gsid) DESC
```

Index:

CREATE INDEX index_playerpid ON public.player USING BTREE (pid)

	Vor der Optimierung	Nach der Optimierung
Planning Time:	0.635 ms	2.165 ms
Execution Time:	894.068 ms	823.029 ms

4. Ausgabe der am meisten gefragten Kategorie, oder alternativ, die Beliebtheit der Kategorien nach Anzahl der Auswahl absteigend sortiert.

Native SQL (zu Testzwecken):

```
SELECT c.name, COUNT(gs.gsid) from Category  AS c, Gamesession AS
gs, Category AS c2, gamesession_category AS gsc
WHERE ((c.cid = c2.cid) AND ((gsc.gamesession_gsid = gs.gsid) AND
(c2.cid = gsc.selectedcategories_cid)))
GROUP BY c.name ORDER BY COUNT (gs.gsid) DESC;
```

Generierte SQL Query:

```
SELECT t0.name, COUNT(t1.GSID) FROM GAMESESSION_CATEGORY t3,
CATEGORY t2, GAMESESSION t1, CATEGORY t0 WHERE ((t0.CID = t2.CID)
AND ((t3.Gamesession_GSID = t1.GSID) AND (t2.CID =
t3.selectedCategories_CID))) GROUP BY t0.name ORDER BY
COUNT(t1.GSID) DESC
```

JPQL Query:

```
SELECT c1.name, COUNT(gs.gsid) FROM Gamesession gs, Category c1
WHERE c1 member of gs.selectedCategories group by c1.name ORDER BY
COUNT(gs.gsid) DESC
```

Index:

CREATE INDEX index_gscategory ON public.gamesession_category USING BTREE (gamesession_gsid)

	Vor der Optimierung	Nach der Optimierung
Planning Time:	5.546 ms	63.548 ms
Execution Time:	3060.920 ms	3002.312 ms

Bereits enthaltene Indizes:

CREATE UNIQUE INDEX sequence_pkey ON public.sequence USING btree (seq_name)
CREATE UNIQUE INDEX category_pkey ON public.category USING btree (cid)
CREATE UNIQUE INDEX answer_pkey ON public.answer USING btree (aid)
CREATE UNIQUE INDEX player_pkey ON public.player USING btree (pid)
CREATE UNIQUE INDEX gamesession_pkey ON public.gamesession USING btree (gsid)
CREATE UNIQUE INDEX game_pkey ON public.game USING btree (gsid)
CREATE UNIQUE INDEX gamesession_question_pkey ON public.gamesession_question USING btree (gamesession_gsid, askedquestions_qid)
CREATE UNIQUE INDEX gamesession_category_pkey ON public.gamesession_category USING btree (gamesession_gsid, selectedcategories_cid
CREATE UNIQUE INDEX game_category_pkey ON public.game_category USING btree (game_gsid, categories_cid, catesssss_cid
CREATE UNIQUE INDEX unq_category_0 ON public.category USING btree (name)

relname	n_live_tup
Gamesession	1000003
Player	10003