Dhruvin Gandhi
Sudhanshu Kulkarni

ZOOKEEPER DESIGN

- We make use of these znodes -
  - /LEADER - stores the request logs (MAX SIZE = 400)
  - /checkpoint - global checkpoint state which stores the application state
  - /(serverID) - stores the counter till which request that replica server has executed
  - /checkpoint_counter - which stores the counter from which the request should be executed after restore
- Initially all the nodes are created in the constructor. The /LEADER node is created whenever the first request hits any of the server.
- /LEADER stores the requests in form of a ArrayList<string>. So the first request will be the 0th index req of this array and SO ON.
- After appending the request, we broadcast the request to all other servers so that other servers know that there has been an update in the /LEADER znode and they can execute the new request.
- Whenever a server receives a request from other server , it means there is an update in /LEADER znode and hence call execute function

Execute function

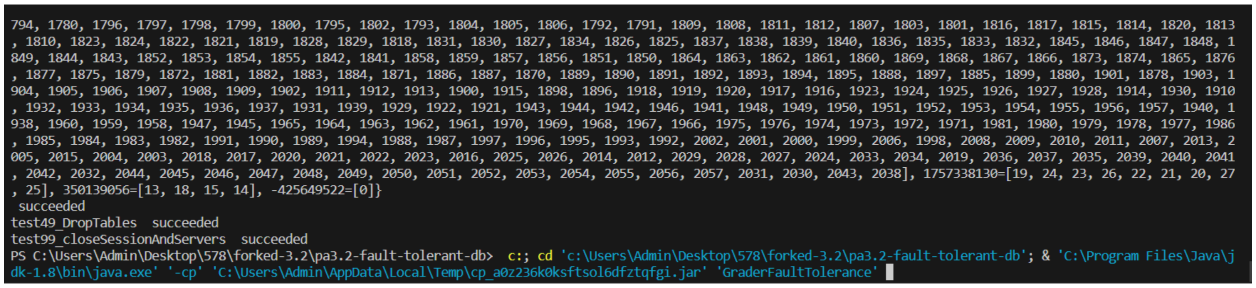
- In this function, we get all the requests stored in /LEADER using the getRequests function().
- We get the current counter stored in the znode(/serverid) . This counter indicates basically till what request has been executed
- We run a for loop from the current counter to allrequestList.size(). And execute the requests.
- After executing the requests, we update the counter to allrequest.size().

Checkpoint function

- Once the List in /LEADER reaches size of 400, we call CHECKPOINT FUNCTION.
- The checkpoint function runs a “SELECT * FROM GRADE” of the **server who got 400th** request and stores the result as a list in global /checkpoint znode.
- Now we get the counter T of this server -> which basically tells till this counter I have executed everything and the checkpoint is stored in the znode.
- So we can remove all the request from /LEADER from 0 to T. (Trimming of logs)
- So basically the leader list hit 400. T is 389 . so we remove all the request from 0 to 389.
- Now only 11 request remain in the /leader node.
- Subsequently we update the counter of all the other servers.

Restore function

- Now if the server wakes up after recovery. We first check if there is anything stored in the global checkpoint znode.
- If there is nothing stored, we change the counter of that server to 0 and call the execute function. So this basically replays all the instruction.
- If there is something stored in checkpoint, we parse it using a regex `Row\\[(-?\\d+), \\[([^\\]]*(?:\\[.*?\\][^\\]]*)*)\\]\\]`
- This regex basically gives the row keys and row values which we store it in a hash map.
- Now we write an insert command which inserts the keys and their corresponding value arrays in the table grade.
- After this we restore the state we get the server checkpoint counter. Basically the counter till which the requests has been executed. We update the server counter to this counter and call the execute function.



```
794, 1780, 1796, 1797, 1798, 1799, 1800, 1795, 1802, 1793, 1804, 1805, 1806, 1792, 1791, 1809, 1808, 1811, 1812, 1807, 1803, 1801, 1816, 1817, 1815, 1814, 1820, 1813
, 1810, 1823, 1824, 1822, 1821, 1819, 1828, 1829, 1818, 1831, 1830, 1827, 1834, 1826, 1825, 1837, 1838, 1839, 1840, 1836, 1835, 1833, 1832, 1845, 1846, 1847, 1848, 1
849, 1844, 1843, 1852, 1853, 1854, 1855, 1842, 1841, 1858, 1859, 1857, 1856, 1851, 1850, 1864, 1863, 1862, 1861, 1860, 1869, 1868, 1867, 1866, 1873, 1874, 1865, 1876
, 1877, 1875, 1879, 1872, 1881, 1882, 1883, 1884, 1871, 1886, 1887, 1870, 1889, 1890, 1891, 1892, 1893, 1894, 1895, 1888, 1897, 1885, 1899, 1880, 1901, 1878, 1903, 1
904, 1905, 1906, 1907, 1908, 1909, 1902, 1911, 1912, 1913, 1900, 1915, 1898, 1896, 1918, 1919, 1920, 1917, 1916, 1923, 1924, 1925, 1926, 1927, 1928, 1914, 1930, 1910
, 1932, 1933, 1934, 1935, 1936, 1937, 1931, 1939, 1929, 1922, 1921, 1943, 1944, 1942, 1946, 1941, 1948, 1949, 1950, 1951, 1952, 1953, 1954, 1955, 1956, 1957, 1940, 1
938, 1960, 1959, 1958, 1947, 1945, 1965, 1964, 1963, 1962, 1961, 1970, 1969, 1968, 1967, 1966, 1975, 1976, 1974, 1973, 1972, 1971, 1981, 1980, 1979, 1978, 1977, 1986
, 1985, 1984, 1983, 1982, 1991, 1990, 1989, 1994, 1988, 1987, 1997, 1996, 1995, 1993, 1992, 2002, 2001, 2000, 1999, 2006, 1998, 2008, 2009, 2010, 2011, 2007, 2013, 2
005, 2015, 2004, 2003, 2018, 2017, 2020, 2021, 2022, 2023, 2016, 2025, 2026, 2014, 2012, 2029, 2028, 2027, 2024, 2033, 2034, 2019, 2036, 2037, 2035, 2039, 2040, 2041
, 2042, 2032, 2044, 2045, 2046, 2047, 2048, 2049, 2050, 2051, 2052, 2053, 2054, 2055, 2056, 2057, 2031, 2030, 2043, 2038], 1757338130=[19, 24, 23, 26, 22, 21, 20, 27
, 25], 350139056=[13, 18, 15, 14], -425649522=[0]}
 succeeded
test49_DropTables  succeeded
test99_closeSessionAndServers  succeeded
PS C:\Users\Admin\Desktop\578\forked-3.2\pa3.2-fault-tolerant-db>  c:; cd 'c:\Users\Admin\Desktop\578\forked-3.2\pa3.2-fault-tolerant-db'; & 'C:\Program Files\Java\j
dk-1.8\bin\java.exe' '-cp' 'C:\Users\Admin\AppData\Local\Temp\cp_a0z236k0ksftsol6dfztqfgi.jar' 'GraderFaultTolerance'
```

- Logic works for more than 15 runs on local.