IMA NEKA LOŠA VEZA (PRIČA O IN-DOUBT DISTRIBUIRANIM TRANSAKCIJAMA)

Zlatko Sirotić, univ.spec.inf.

Istra informatički inženjering d.o.o.

Pula

Uvod

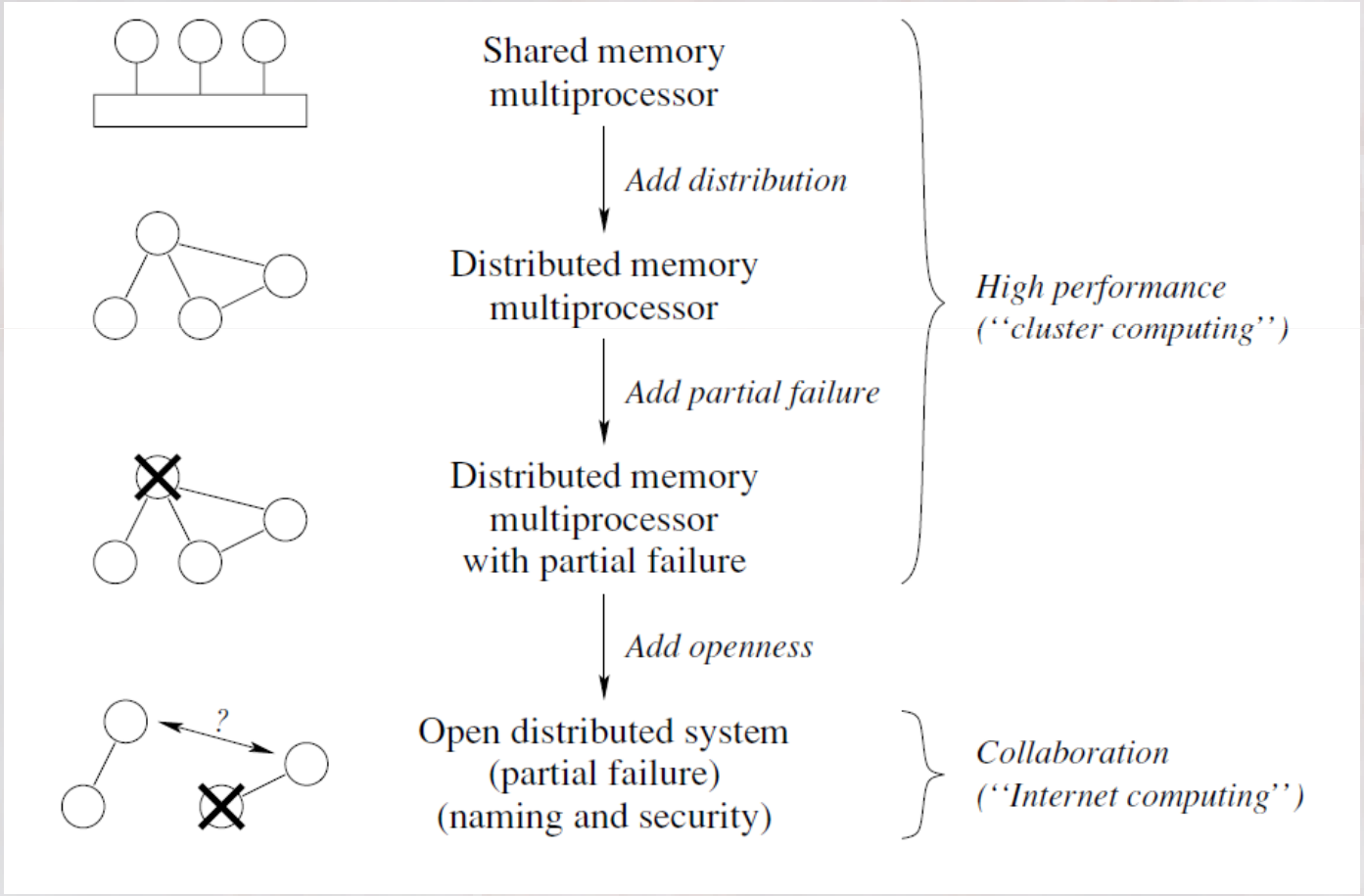
- Najlakše i najbolje je ako možemo raditi samo sa centraliziranom bazom podataka. No, ponekad nužno moramo raditi sa **distribuiranom bazom poda**taka.
- Kod distribuirane baze podataka često se radi **replikacija podataka**, koja može biti **asinkrona ili sinkrona**. Asinkrona je u pravilu brža, ali ne osigurava konzistentnost podataka.
- Sinkrona replikacija je jedna vrsta **distribuirane transakcije**. Distribuirana transakcija koristi **dvofazni commit protokol**.
- Kod dvofaznog commit protokola postoji **period u kojem je transakcija osjetljiva** na pad (nekog) servera baze ili veze. U slučaju greške, **transakcija postaje in-doubt**.
- In-doubt transakcije najčešće baza rješava sama, ali **ponekad ih moramo ručno razriješiti**.

Teme

- Distribuirani sustavi
- Arhitektura Oracle baze
- Transakcije
- Distribuirana baza podataka
- Replikacija podataka
- Distribuirane transakcije
- Dvofazni commit protokol
- In-doubt distribuirane transakcije

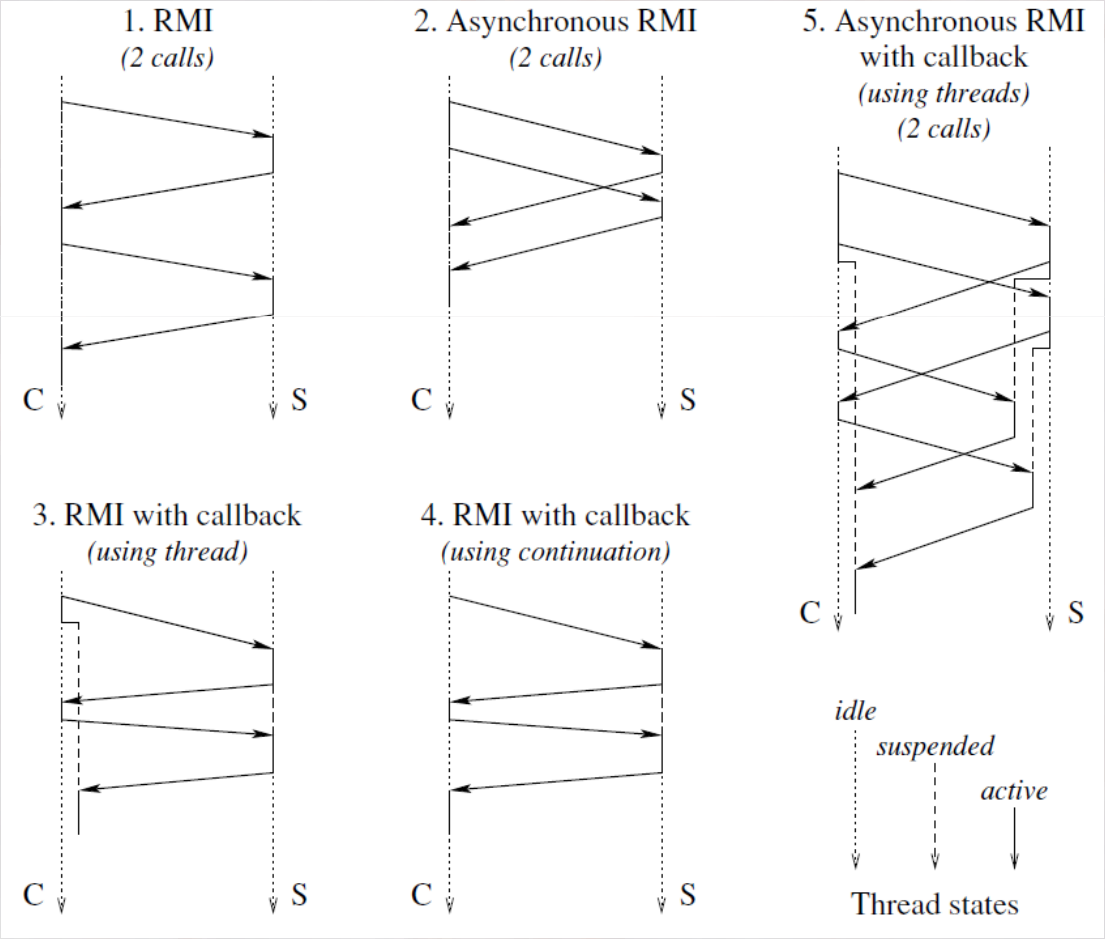
Distribuirani sustavi

- Taksonomija distribuiranih sustava:



Distribuirani sustavi – neki jednostavni protokoli

- Asinkroni sustavi su brži, ali teško osiguravaju konzistentnost.



Arhitektura Oracle baze

- Arhitekturu Oracle sustava čine **dva glavna dijela**:
- **Baza podataka** (BP), koju čine različite vrste datoteka; **najvažnije su datoteke podataka**, no postoji i desetak drugih vrsta datoteka, od kojih su posebno važne **Redo Log** datoteke, koje se dijele na Online i Archive; Online Redo Log datoteke služe za oporavak baze u slučaju pada sustava (npr. programske greške ili nestanaka struje), dok Archive Redo Log datoteke služe za uspostavu prijašnjeg stanja, zajedno sa backup datotekama, u slučaju kvara diska;
- **Instanca** (jedna ili više) baze podataka, koju čine memorijske strukture i procesi u memoriji; od memorijskih struktura, naročito su zanimljivi Block Buffer Cache i Redo Bufer.

Online Redo Log datoteke

- Online Redo Log čine minimalno dvije datoteke (a preporučljivo je da postoje barem tri) koje se koriste u krug – kada se prva napuni, Oracle počinje pisati u drugu, a kada se druga napuni Oracle se vraća na prvu.
- Kada su podaci zapisani u Online Redo Log može doći do pada sustava (to nije kvar medija) prije nego Oracle te podatke upiše u prave tablice podataka. No, **nakon što se Oracle sustav ponovno podigne, pročitat će Online Redo Log i upisati podatke u prave tablice**.
- Naravno, pitanje je da li ti podaci zaista trebaju biti trajno zapisani u tablicu podataka – to ovisi o tome da li su oni commit-irani.

UNDO tablespace

- UNDO tablespace **služi za elimininiranje (iz tablica podataka) promjena koje nisu commit-irane**, kod oporavka sustava nakon pada.
- **Služi i za omogućavanje višekorisničkog rada**. Budući da se često ne-commit-irani podaci moraju spremati na disk, pitanje je otkuda sesija može čitati stare podatke (prije promjene) – oni se nalaze u UNDO tablespaceu.
- Postojanje UNDO tablespacea (ili neke slične strukture u nekom drugom RSUBP sustavu) omogućava da mijenjanje podataka ne utječe na čitanje podataka, tj. **mijenjanje podataka ne sprečava da se ti podaci istovremeno i čitaju**. U RSUBP sustavima koji nemaju nešto slično kao što je UNDO tablespace, mijenjanje podataka utječe na čitanje.

Osobine Oracle transakcije

- Sesija BP **ne vidi promjene koje je napravila druga sesija**, dok druga sesija ne napravi COMMIT (ili ROLLBACK). Međutim, sesije nisu nezavisne, jer zaključavanje redaka u jednoj sesiji utječe na drugu sesiju koja pokušava ažurirati redak koji je zaključala prva sesija.
- Kada sesija izvršava DML naredbu, **automatski se zaključa redak**. Redak se može otključati tek na kraju transakcije (COMMIT ili ROLLBACK), ili pomoću ROLLBACK TO SAVEPOINT. Međutim, redovi koji su otključani sa ROLLBACK TO SAVEPOINT ostaju zaključani za one sesije koje su već prije toga pokušale izvršiti DML na tim redovima.
- Redovi se mogu zaključati i sa SELECT ... FOR UPDATE. Pritom postoji i opcija NOWAIT / WAIT n (sekundi).

Osobine Oracle transakcije - nastavak

- Transakcija može ili u cijelosti uspjeti (COMMIT), ili biti u cijelosti poništena (ROLLBACK).
- Pritom se mogu desiti **tri tipa grešaka**:

- narušen **uvjet na tip stupca**, npr. upis 100 u NUMBER (2)
- narušeno **deklarativno integritetno ograničenje** (PK, UK, FK, CK, NOT NULL)
- narušeno **proceduralno ograničenje** (okidači baze).

- Ako se desila greška na bazi koja nije obrađena, a početak je DML naredba, **poništavaju se svi efekti naredbe**.
- Ako se desila greška na bazi, a početak je poziv procedure sa strane klijenta (npr. Forms), ili poziv sa strane druge baze (udaljena procedura), **poništavaju se svi efekti procedure**.

Distribuirana baza podataka

- Sustav distribuiranih baza podataka (ili jednostavnije - distribuirana baza podataka) je sustav od dvije ili više baza podataka koje bi aplikacijama (korisničkim programima) **trebale izgledati kao jedna jedinstvena baza podataka**.
- Date je u svojoj knjizi postavio "temeljni princip za distribuirane baze podataka", a to je: "Za korisnika, distribuirani sustav (baza podataka) treba izgledati potpuno isto kao ne-distribuirani sustav".
- Na temelju tog principa, Date u knjizi daje **12 zahtjeva koje distribuirana baza mora zadovoljiti**. Činjenica je da je 100%-tno zadovoljenje svih tih zahtjeva gotovo nemoguće, ali ti zahtjevi služe kao ideal prema kojemu bi trebale težiti konkretne implementacije distribuiranih baza podataka.

Database link

- Oracle baza podataka zadovoljava Dateove zahtjeve u velikoj mjeri. U zadovoljavanju tih zahtjeva veliku ulogu igra nepostojanje "pravog" globalnog rječnika podataka u Oracle bazi, jer svaka Oracle baza ima svoj lokalni rječnik podataka.
- Za čuvanje informacija o udaljenim objektima, Oracle baza podataka koristi tzv. "database link". Može se reći da je database link objekt baze podataka koji definira jednosmjernu vezu baze podataka na drugu bazu podataka.
- Slijedi primjer definiranja privatnog database linka:

```
CREATE DATABASE LINK neki_link
   CONNECT TO shemaY IDENTIFIED BY zaporka
   USING "bazaB";
```

Udaljeni (remote) upit i DML naredba, distribuirani upit i DML naredba, distribuirana transakcija

- **Udaljeni upit**

```
SELECT * FROM neka_tablica@neki_link;
```

- **Distribuirani upit**

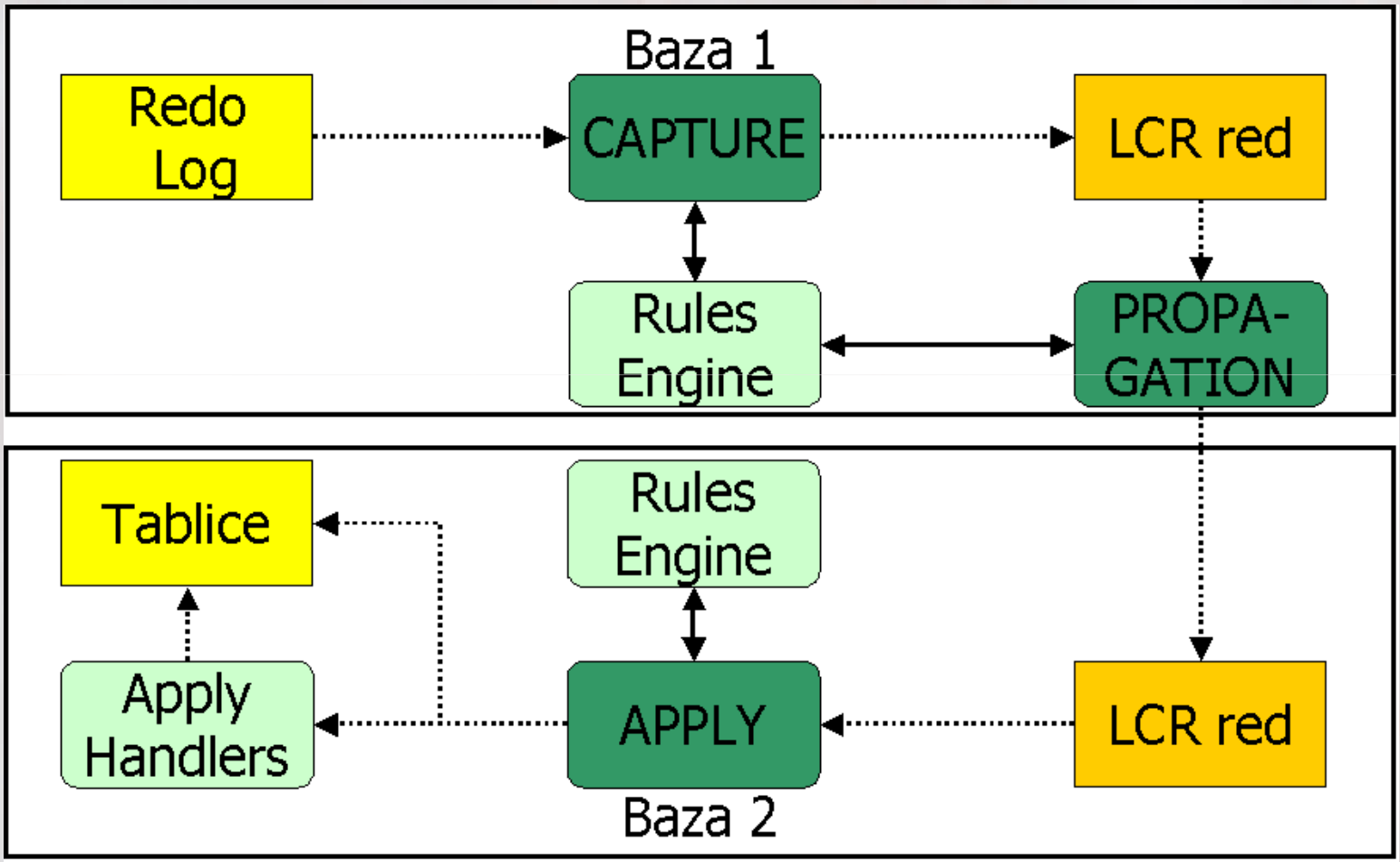
```
SELECT * FROM lokalna_tab,
   udaljena2@baza2, udaljena3@baza3 WHERE ...
```

- Osim udaljenog upita / distribuiranog upita, Oracle podržava i udaljene / distribuirane DML naredbe.
- **Distribuirana transakcija** je ona transakcija u toku koje se ažuriraju (tj. unose, mijenjanju ili brišu) redovi iz barem dvije tablice koje se nalaze u različitim bazama podataka. Distribuirani upit ne mora sadržavati distribuiranu DML naredbu, niti distribuirani upit.

Replikacija podataka - Oracle varijante

- Može se reći da je replikacija **proces kreiranja kopija podataka** (i metapodataka) i usklađivanja (sinkroniziranja) podataka (i metapodataka) **u bazama koje čine distribuiranu bazu podataka**. Sve promjene koje se naprave na jednoj bazi (distribuirane baze) moraju se odraziti i na kopije podataka u ostalim bazama (koje imaju te kopije).
- Oracle ima više varijanti replikacije. Postoji **bazična replikacija**, koja je raspoloživa i u Standard ediciji baze i koja omogućava samo asinkronu replikaciju, sa jednom master bazom. **Napredna replikacija**, za razliku od bazične, omogućava i sinkronu replikaciju, multi-master replikaciju i proceduralnu replikaciju. Novija Oracle replikacija je tzv. Oracle **Streams**, a ona je replikacija temeljena na logu.

Oracle Streams procesi



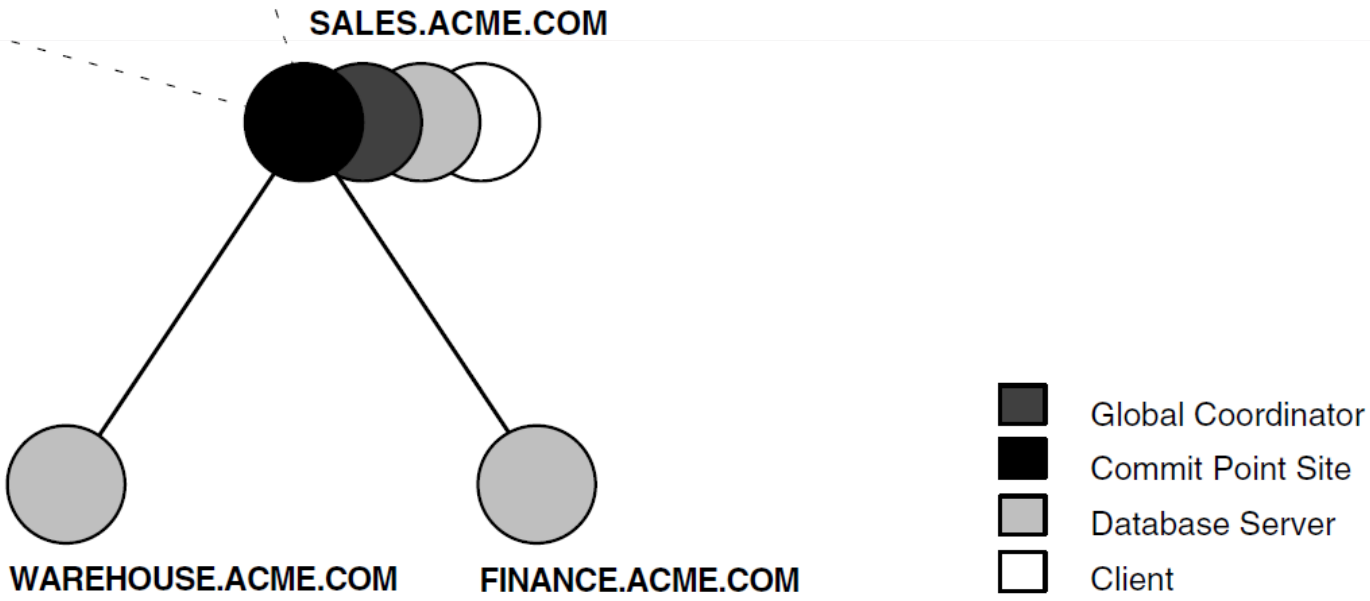
Osobine Oracle varijanti replikacije (i našeg rješenja sinkrone replikacije u Standard ediciji baze)

Tablica 1: Osnovne osobine Oracle i naše replikacije

Replikacija / Osobine	MV	UMV	Multi-Master	Proce-dural	Streams	Naša
Transakcijska ili Log bazirana	T	T	T	T	L	T
Sinkrona ili Asinkrona	A	A	S / A	S / A	A	S
Retčana ili Proceduralna	R	R	R	P	R	R
Jednosmjerna ili Dvosmjerna	J	D	D	D	D	J

Primjer distribuirane transakcije

```
INSERT INTO orders...;
UPDATE inventory @ warehouse...;
UPDATE accts_rec @ finance...;
.
COMMIT;
```



Učesnici u distribuiranoj transakciji

- **Klijent** (client): baza koja referencira podatke na drugim bazama.
- **Server baze podataka** (database server): bilo koja baza podataka koja sadrži podatke koje druge baze (uključujući samu sebe) referenciraju;
- **Lokalni koordinator** (local coordinator): baza koja mora referencirati podatke na drugim bazama da bi završila svoj dio distribuirane transakcije. Vidi samo svoje neposredne susjede. Komunicira sa svojim "podređenim" susjednim bazama, šalje im upite i prima od njih rezultate (ali i informacije o statusu transakcije), pa onda te rezultate šalje "nadređenim" susjednim bazama (koje su upite inicirale).

Učesnici u distribuiranoj transakciji - globalni koordinator

- **Globalni koordinator** (global coordinator): baza koja je izvorište distribuirane transakcije. Aplikacija koja je pokrenula distribuiranu transakciju direktno je vezana za nju. Radi sljedeće operacije za vrijeme distribuirane transakcije:

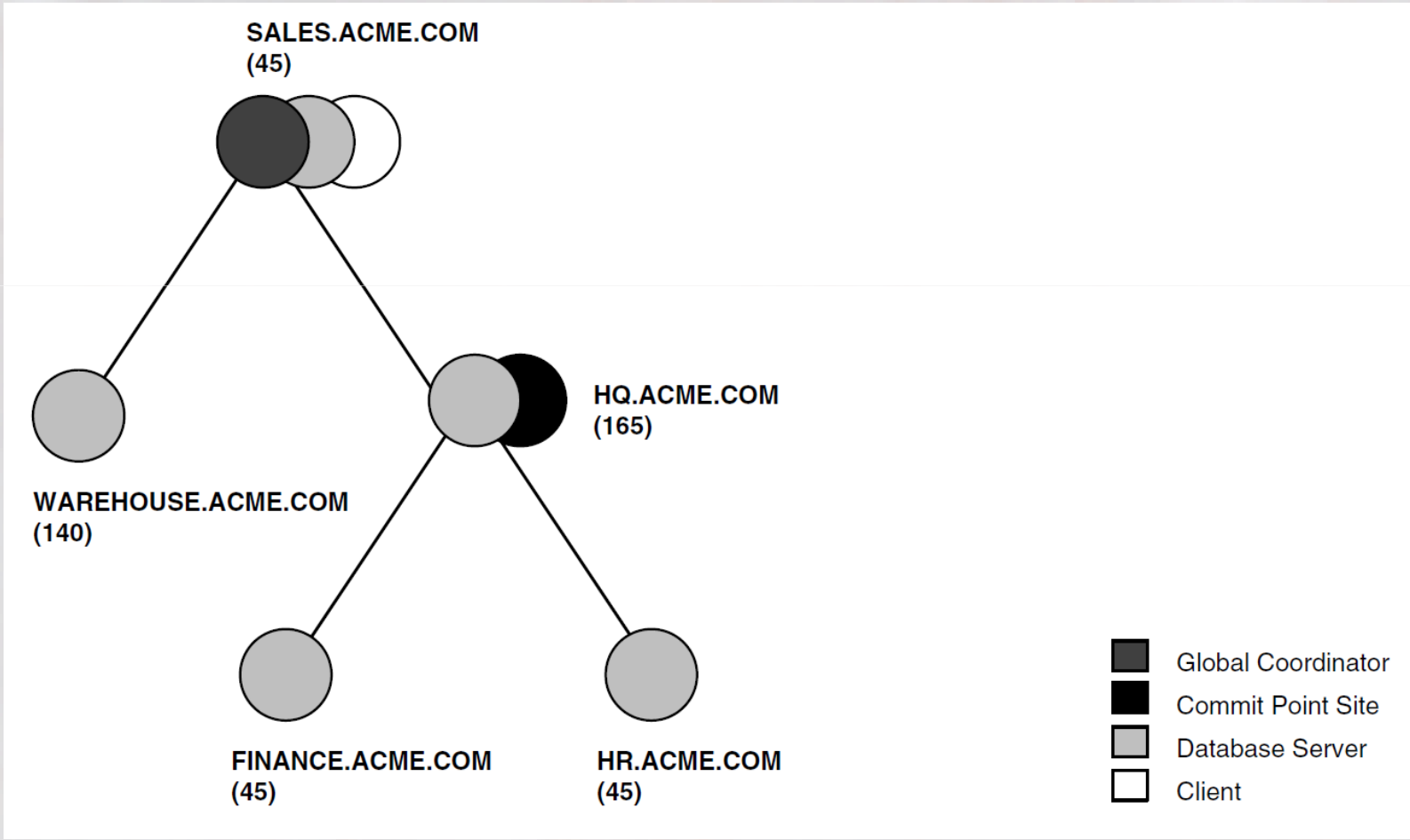
  1. Šalje SQL naredbe ili pozive udaljenih procedura na određenu bazu.
  2. Šalje svim bazama sa kojima je direktno povezan, osim commit point site bazi, naredbe da se pripreme.
  3. Ako sve baze odgovore da su se pripremile, šalje commit point site bazi naredbu da izvrši commit.
  4. Ako neka baza odgovori sa abort, šalje svim bazama naredbu da izvrše globalni rollback.

Učesnici u distribuiranoj transakciji - commit point site

- **Commit point site**: to je baza koja inicira globalni commit ili globalni rollback, kada dobije takvu instrukciju od globalnog koordinatora. Baza koja će biti commit point site **trebala bi biti ona koja čuva najvažnije podatke**. Razlikuje se od ostalih baza u distribuiranoj transakciji po sljedeće dvije karakteristike:
  1. Ona **nikada ne ulazi u pripravno stanje** (prepared state). Za razliku od toga, ostale baze kod greške ostaju u pripremnom stanju, držeći (potrebne) zaključane retke, dok god se ne riješi **in-doubt transakcija**.
  2. Ona **izvršava commit prije svih ostalih baza**. Nakon toga je distribuirana transakcija commit-irana, jer se globalni koordinator brine da sve druge baze isto izvrše commit.

Izbor commit point site baze – na temelju parametra baze COMMIT_POINT_STRENGTH

- U sljedećem primjeru, Commit point site je baza HQ:



Oracle dvofazni commit protokol ima tri faze; 1. faza je faza pripreme (prepare phase)

- U **fazi pripreme**, globalni koordinator traži od ostalih baza, osim commit point site baze, da pređu u **pripravno stanje** (prepared state). Transakcija je od tada u **osjetljivom stanju**.
- Pripremljene baze stavljaju distribuirani lokot na sve svoje modificirane tablice. **Taj lokot sprečava čak i čitanje podataka (što Oracle baza inače ne radi), pa u slučaju problema podaci ostaju zaključani i za čitanje!**
- Inače, baza može odgovoriti na sljedeća tri načina:
  1. **Prepared**: baza javlja da je pripravna.
  2. **Read-only**: baza javlja da ona nije radila nikakav DML, pa ne treba ići u pripravno stanje.
  3. **Abort**: baza javlja da se ne može pripremiti, otključava sve svoje zaključane retke i radi lokalni ROLLBACK.

Faza potvrde (commit phase) i faza zaboravljanja (forget phase)

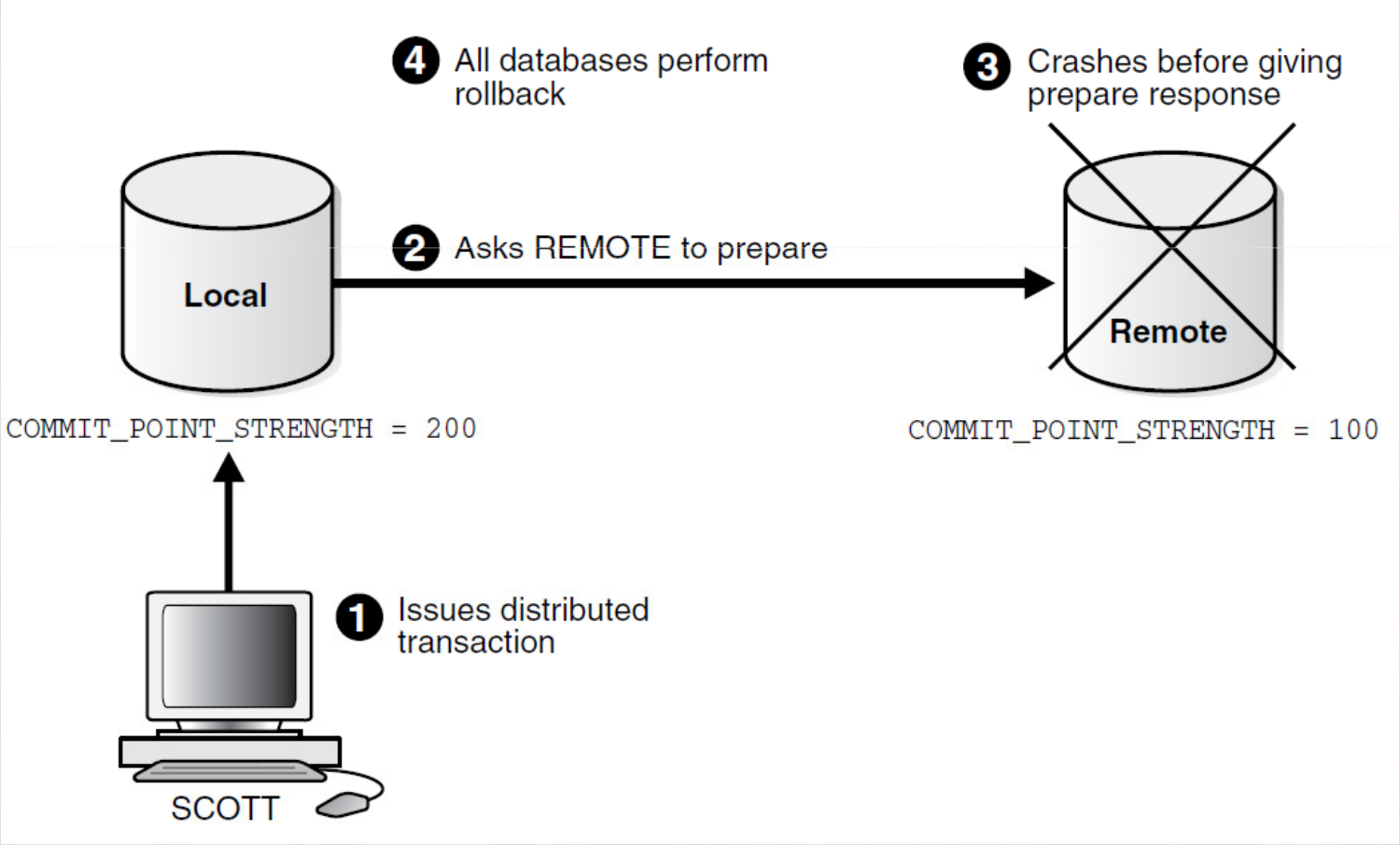
- Ako globalni koordinator od barem jedne baze dobije odgovor **Abort**, on šalje svim bazama ROLLBACK. Ako od svih baza dobije odgovor Prepared, **započinje fazu potvrde**:
  - Šalje commit point site bazi naredbu da izvrši COMMIT. Commit point site baza izvršava COMMIT i obavještava globalnog koordinatora.
  - Globalni koordinator i lokalni koordinatori šalju naredbe svim svojim podređenim bazama, tražeći od njih da naprave COMMIT. Ostale baze rade lokalni COMMIT.
- **U fazi zaboravljanja**, globalni koordinator kaže commit point site bazi da zaboravi transakciju, tj. da izbriše stanja o transakciji koja ona vodi kod sebe. Globalni koordinator nakon toga briše i kod sebe informacije o distribuiranoj transakciji.

In-doubt transakcija

- U slučaju da se u bilo kojoj fazi desi greška (preciznije, nakon što transakcija postane "osjetljiva", tj. nakon što barem jedna baza završi fazu pripreme), distribuirana transakcija postaje in-doubt. Greške mogu biti:
  - **Padne računalo** na kojem radi Oracle baza.
  - **Prekine se veza** između dvije ili više baza koje sudjeluju u distribuiranoj transakciji.
  - Desi se neki **softverski problem**.
- **RECO proces baze** najčešće **automatski rješava in-doubt** transakciju nakon što se podigne računalo, uspostavi veza, odnosno riješi softverski problem. No, dok RECO proces ne riješi problem, ostaju zaključani podaci modificiranih tablica, i to (kako je već rečeno) ne samo za pisanje, već i za čitanje.

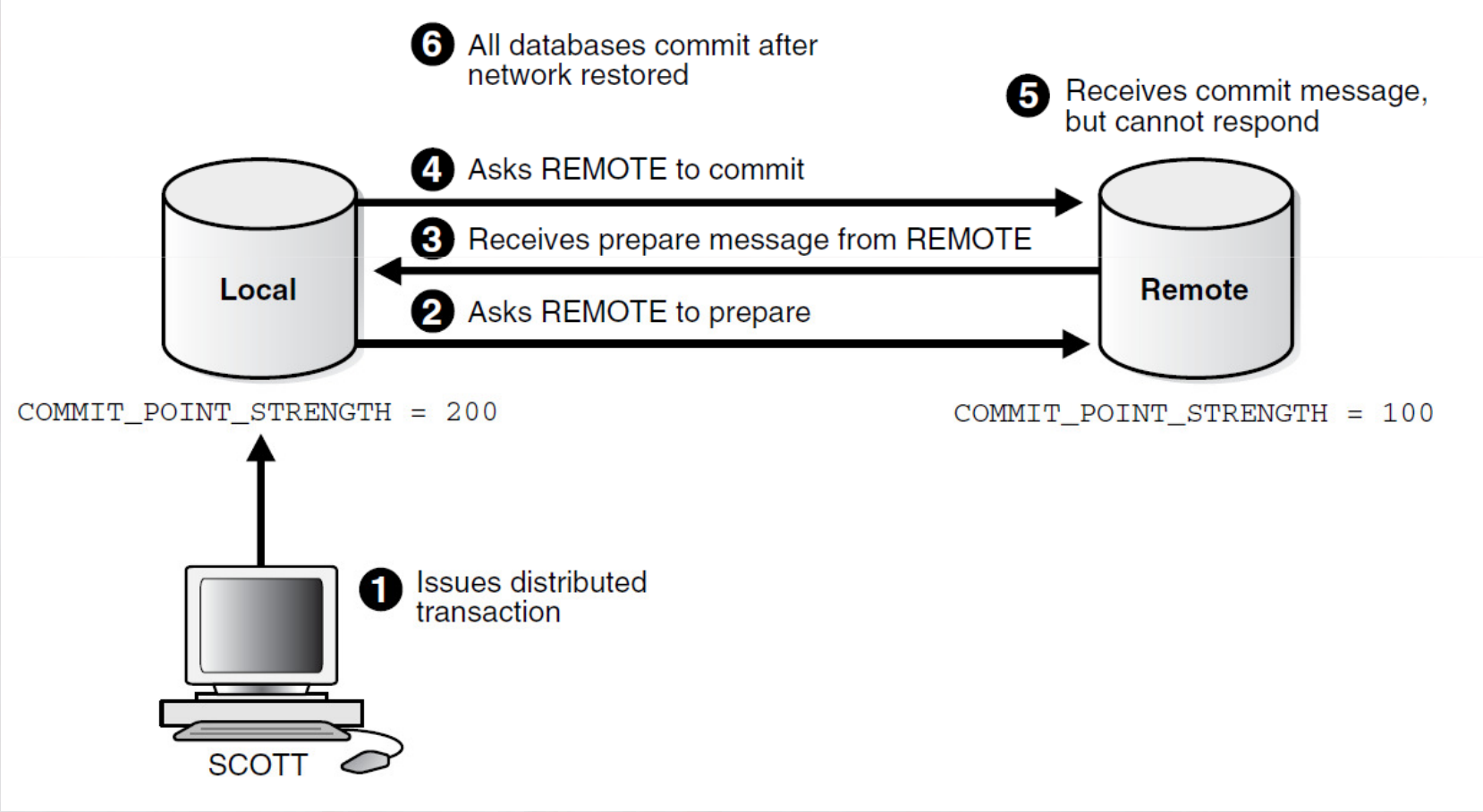
In-doubt transakcija nastala u fazi pripreme

- Ovdje nije veliki problem, jer samo baza Remote ima podatke zaključane za pisanje i čitanje, a rješavanje je automatsko:



In-doubt transakcija nastala u fazi potvrde

- Niti ovdje nije veliki problem, iz istih razloga kao prije:



Kada ručno rješavati in-doubt transakciju

- Treba ručno razriješiti in-doubt transakciju ako je:
  1. In-doubt transakcija **zaključala kritične podatke** ili undo segmente.
  2. Ili se pad računala, prekid veze, ili softverski problem **ne mogu riješiti u kratkom vremenu**.
- Ručno rješavanje in-doubt transakcije može biti kompleksno. Glavni postupak **sastoji se od sljedećih koraka**:
  1. Pronaći identifikacijski broj in-doubt transakcije.
  2. Postaviti upit nad sistemskim view-ovima **DBA_2PC_PENDING** i **DBA_2PC_NEIGHBORS**, kako bismo odredili da li su baze koje sudjeluju u transakciji, commit-irale.
  3. Ako je potrebno, forsirati commit sa **COMMIT FORCE**, odnosno rollback sa **ROLLBACK FORCE** naredbom.

ORA greške kod in-doubt transakcije

- Identifikacija in-doubt transakcije može se vidjeti već kod prve pojave greške, kod **korisnika koji radi COMMIT**:

```
ORA-02050: transaction ID rolled back,
     some remote dbs may be in-doubt
ORA-02053: transaction ID committed,
     some remote dbs may be in-doubt
ORA-02054: transaction ID in-doubt
```

- Zapravo je veća šansa da će grešku javiti **korisnik kojemu su podaci ostali zaključani** ne samo za pisanje, već i za čitanje:

```
ORA-01591: lock held by in-doubt
     distributed transaction ID
```

View-ovi DBA_2PC_PENDING i DBA_2PC_NEIGHBORS

- Sada kada znamo identifikaciju in-doubt transakcije, možemo analizirati view-ove DBA_2PC_PENDING i DBA_2PC_NEIGHBORS.
- View DBA_2PC_PENDING prikazuje **sve in-doubt distribuirane transakcije**. Prazan je ako ih nema, a nakon što se in-doubt transakcije riješe, isto bi se trebao automatski isprazniti (u većini slučajeva).
- Naročito su zanimljiva dva njegova stupca, STATE i MIXED. Stupac **MIXED** trebao bi uvijek sadržavati vrijednost NO, osim u slučaju ako smo greškom na nekoj bazi forsirali commit, a na drugoj rollback, pa je **distribuirana transakcija ostala u "miksanom" stanju**, što baza sama prepoznaje i stavlja vrijednost YES.

Vrijednosti stupca STATE u DBA_2PC_PENDING

- **Collecting**: ovu vrijednost normalno sadrže samo baze koje su **globalni ili lokalni koordinatori**; To znači da te baze prikupljaju informacije sa ostalih servera baze, prije nego odluče da li će se i one pripremiti.
- **Prepared**: znači da je baza pripremljena (što znači i da je zaključala retke modificiranih podataka i za pisanje i za čitanje), ali mogla je to javiti, ili ne stići javiti svom koordinatoru. Baza čeka na poruku da napravi commit.
- **Committed**: znači da je ova baza commit-irala.
- **Forced commit** / **Forced rollback** : znači da je transakciju, koja je prije bila u statusu Prepared, administrator ručno riješio sa **COMMIT FORCE** / **ROLLBACK FORCE**.

Što napraviti kod ručnog rješavanja in-doubt transakcije

STATE Column	State of Global Transaction	State of Local Transaction	Normal Action	Alternative Action
Collecting	Rolled back	Rolled back	None	PURGE_LOST_DB_ENTRY (only if autorecovery cannot resolve transaction)
Committed	Committed	Committed	None	PURGE_LOST_DB_ENTRY (only if autorecovery cannot resolve transaction)
Prepared	Unknown	Prepared	None	Force commit or rollback
Forced commit	Unknown	Committed	None	PURGE_LOST_DB_ENTRY (only if autorecovery cannot resolve transaction)
Forced rollback	Unknown	Rolled back	None	PURGE_LOST_DB_ENTRY (only if autorecovery cannot resolve transaction)
Forced commit	Mixed	Committed	Manually remove inconsistencies then use PURGE_MIXED	-
Forced rollback	Mixed	Rolled back	Manually remove inconsistencies then use PURGE_MIXED	-

Simuliranje greške kod in doubt-transakcije

- Kako bi se omogućilo vježbanje ručnog rješavanja in-doubt transakcija, Oracle je omogućio da se simuliraju greške koje dovode do in-doubt transakcija.

- To se radi tako da se umjesto COMMIT naredbe upotrijebi naredba:

```
COMMIT COMMENT 'ORA-2PC-CRASH-TEST-n';
```

- pri čemu *n* može imati vrijednost od 1 do 10, a ima efekt koji je prikazan u sljedećoj tablici:

Efekt vrijednosti *n* kod naredbe COMMIT COMMENT 'ORA-2PC-CRASH-TEST-*n*'

n	Effect
1	Crash commit point after collect
2	Crash non-commit-point site after collect
3	Crash before prepare (non-commit-point site)
4	Crash after prepare (non-commit-point site)
5	Crash commit point site before commit
6	Crash commit point site after commit
7	Crash non-commit-point site before commit
8	Crash non-commit-point site after commit
9	Crash commit point site before forget
10	Crash non-commit-point site before forget

Isključivanje / uključivanje RECO pozadinskog procesa baze

- Kako bi se spriječilo RECO proces baze da na brzinu automatski riješi simulirani in-doubt problem, RECO se može isključiti, pomoću:

```
ALTER SYSTEM DISABLE DISTRIBUTED RECOVERY;
```

- Ponovno uključivanje radi se sa:

```
ALTER SYSTEM ENABLE DISTRIBUTED RECOVERY;
```

Zaključak

- Ponekad nužno moramo raditi sa **distribuiranom bazom podataka i distribuiranim transakcijama**.
- Distribuirana transakcija koristi **dvofazni commit protokol**. Kod njega postoji **period u kojem je transakcija osjetljiva** na pad (nekog) servera baze ili veze. U tom slučaju (ako se desi greška) **transakcija postaje in-doubt**.
- U **fazi pripreme,** baze stavljaju distribuirani lokot na sve modificirane tablice. **Taj lokot sprečava čak i čitanje podataka!** Ako to traje kratko, nije problem. No, ako transakcija postane in-doubt, može se desiti da duže vrijeme drži zaključane podatke (čak i za čitanje).
- In-doubt transakcije najčešće baza rješava sama, ali **ponekad ih moramo ručno razriješiti**.