Bancos de Dados Distribuídos

UPIS Graduação • Pós • MBA
CADA DIA MELHOR
CADA DIA MAIS UP

- Tipos de distribuição
  - Particionamento
  - Replicação
- Distribuição no Oracle
  - Database link e sinônimo
  - Materialized View
- Transações distribuídas
  - Protocolo 2PC

O que é distribuído?

- Banco de Dados Distribuído (BDD)
  - Uma coleção de bancos de dados, interrelacionados logicamente
  - Através de uma rede de computadores
- Sistema Gerenciador de Banco de Dados Distribuído (SGBDD)
  - Software que gerencia um BDD
  - Que provê um mecanismo de acesso que torna a distribuição transparente para os usuários

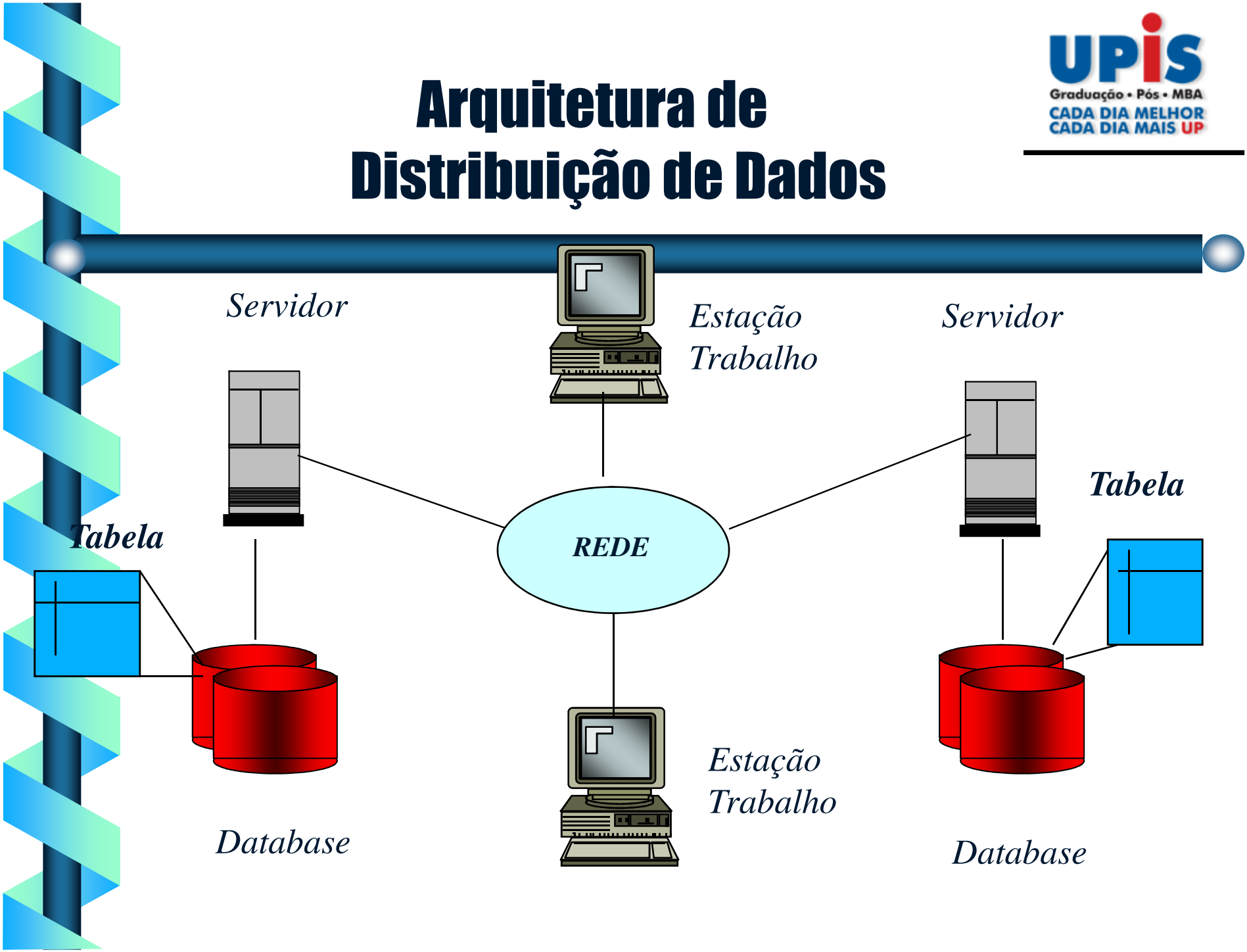
Tipos de SGBDD

- Sistema Gerenciador de Banco de Dados Distribuído Homogêneo
  - Todos os sites usam o mesmo SGBD
  - Mais fácil de projetar e gerenciar
- Sistema Gerenciador de Banco de Dados Distribuído Heterogêneo
  - Sites usam diferentes SGBD's
  - Mais comum de ocorrer
  - Mais complexo de projetar e gerenciar

Requisitos para definição de arquitetura

- Devem ser avaliados os seguintes requisitos
  - Disponibilidade dos Dados
  - Integridade do Acervo
  - Performance de Acesso
  - Gestão do Ambiente
  - Segurança quanto ao acesso
  - Infra-estrutura de Rede
  - Transparência da distribuição

Arquitetura de Distribuição de Dados



Distribuição por Particionamento de Dados

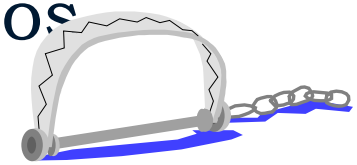
- Particionamento Horizontal
  - Selecionam-se algumas linhas da tabela
- Particionamento Vertical
  - Selecionam-se algumas colunas da tabela
- Particionamento Híbrido
  - Selecionam-se algumas colunas de algumas linhas da tabela

Distribuição por Particionamento/Fragmentação de Dados

Particionamento - Vantagens

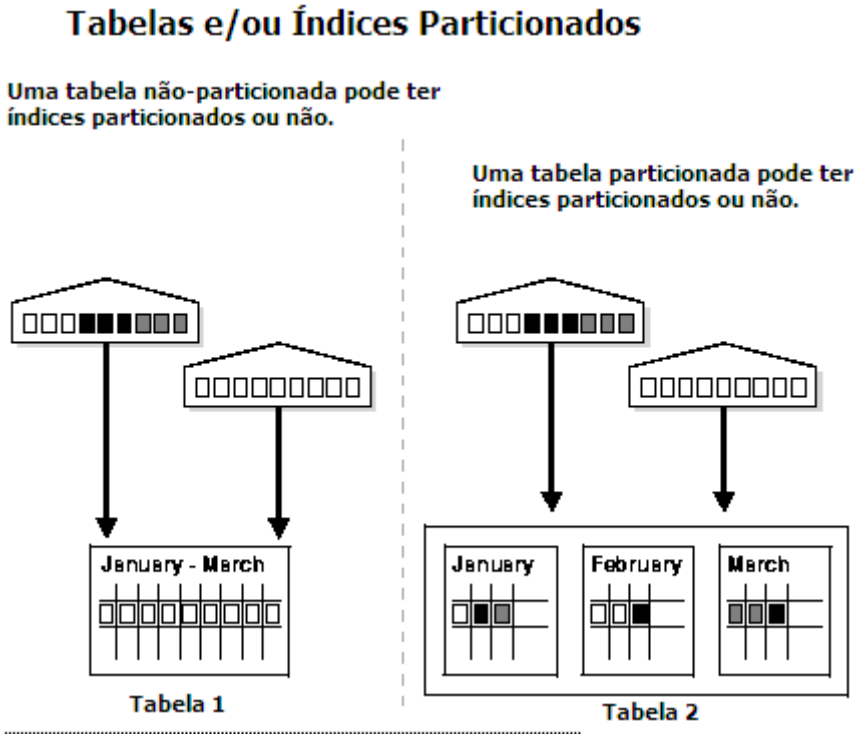
- disponibilidade dos dados
- menor volume para manipulação
- performance de aplicativos

Particionamento - Pontos Críticos



- gestão de diversos ambientes
- consistência de dados e integridade
- complexidade de implementação
- complexidade de recuperação

Particionamento no Oracle



- Particionamento é muito importante para grandes tabelas ou índices, pois permite sua decomposição em pedaços menores chamados partições.

  Comandos de DML não precisam ser alterados para acessarem tabelas particionadas.

- Comandos de DDL podem fazer referência a partições individuais, com isso se facilita a manutenção de objetos muito grandes, aumentado-se sua disponibilidade e melhorando-se a performance.

Particionamento no Oracle

UPIS
Graduação • Pós • MBA
CADA DIA MELHOR
CADA DIA MAIS UP

- **Range Partitioning Example**
- CREATE TABLE sales_range (salesman_id NUMBER(5), salesman_name VARCHAR2(30),

```
sales_amount NUMBER(10),
sales_date DATE)
PARTITION BY RANGE(sales_date)
( PARTITION sales_jan2000 VALUES LESS
    THAN(TO_DATE('02/01/2000','DD/MM/YYYY')),
PARTITION sales_feb2000 VALUES LESS
    THAN(TO_DATE('03/01/2000','DD/MM/YYYY')),
PARTITION sales_mar2000 VALUES LESS
    THAN(TO_DATE('04/01/2000','DD/MM/YYYY')),
PARTITION sales_apr2000 VALUES LESS
    THAN(TO_DATE('05/01/2000','DD/MM/YYYY')), )
```

Particionamento no Oracle

UPIS
Graduação • Pós • MBA
CADA DIA MELHOR
CADA DIA MAIS UP

- **List Partitioning Example**
- CREATE TABLE sales_list (salesman_id NUMBER(5),
  salesman_name VARCHAR2(30),
  sales_state VARCHAR2(20),
  sales_amount NUMBER(10),
  sales_date DATE)
  PARTITION BY LIST(sales_state)
  (PARTITION sales_west VALUES IN ('California', 'Hawaii'),
  PARTITION sales_east VALUES IN ('New York', 'Virginia',
  'Florida'),
  PARTITION sales_central VALUES IN ('Texas', 'Illinois'),)

Particionamento no Oracle

UPIS
Graduação • Pós • MBA
CADA DIA MELHOR
CADA DIA MAIS UP

- **Hash Partitioning Example**
- CREATE TABLE sales_hash

```
(salesman_id NUMBER(5),
salesman_name VARCHAR2(30),
sales_amount NUMBER(10),
week_no NUMBER(2))
PARTITION BY HASH(salesman_id)
PARTITIONS 4
STORE IN (data1, data2, data3, data4)
```

Distribuição por Replicação de Dados

UPIS
Graduação • Pós • MBA
CADA DIA MELHOR
CADA DIA MAIS UP

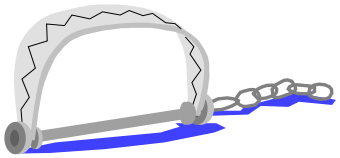
- Replicação
  - acervo repetido em mais de um servidor
  - necessidade de tratamento de redundâncias
  - Pontos importantes a considerar:
    - Quais dados replicar?
    - Qual a frequência de atualização das réplicas?
    - Como resolver e/ou evitar conflitos de atualização nos múltiplos sites de replicação?

Distribuição por Replicação de Dados

Replicação - Vantagens

- maior disponibilidade dos dados
- performance de aplicativos

Replicação - Pontos Críticos

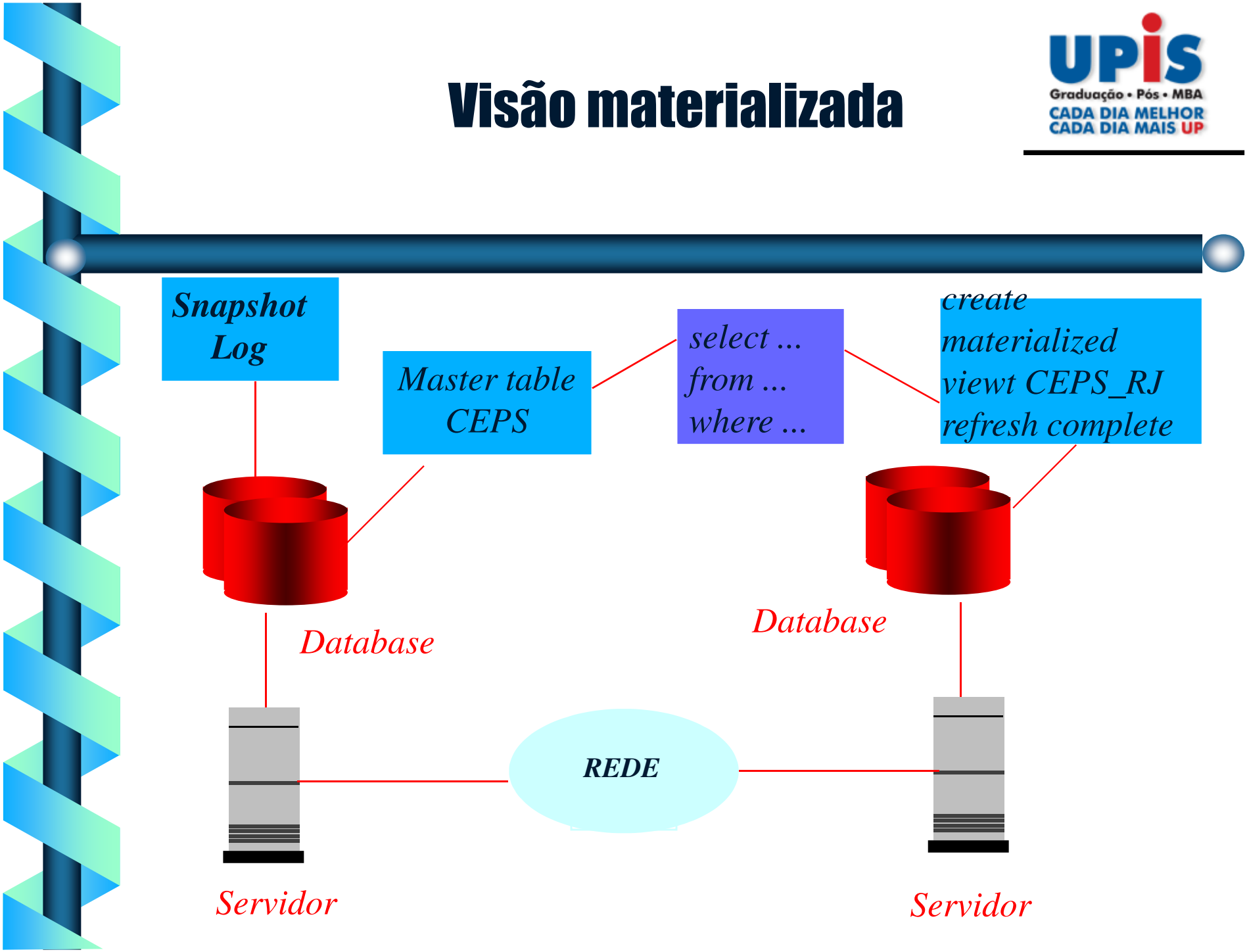


- gestão de diversos ambientes
- consistência de dados: sincronismo
- complexidade de implementação
- Dados podem acabar espalhados de forma não controlada

Replicação no Oracle

- ***Materialized View***
  - Visão materializada, que replica toda ou parte de uma ou mais tabelas, gravando os dados em disco
  - *Simple*, cópia completa ou parcial de uma tabela
  - *Complex*, baseado em mais tabelas
  - Reflete um estado recente da tabela mestre
  - Também pode referenciar *views*
  - Pode ser atualizável (*updatable*) ou não (*read-only*)
  - No *select* podem ser usadas cláusulas *where* e *group by*
  - É possível seus dados serem *refreshed* por completo, ou então com o *snapshot log são* refletidas só as atualizações

Visão materializada



Consultas Remotas

- Database link
  - Estabelece o nome de serviço a ser usado
  - Pode especificar o usuário a ser conectado
  - Pode ser criado como público ou privado
  - É usado como sufixo ao nome da tabela
  - Exemplo
    - create public database link SDC_LINK
      connect to SDCCON identified by SDCSENHA
      using ‘sdc’;
    - select * from MUNICIPIOS@SDC_LINK
      where sg_uf = ‘RJ’;

Two-Phase Commit

- 2PC – *Commit* em Duas Fases
  - Características
    - Permite grupo de transações entre diversos nós, ser tratado como uma só unidade lógica
    - Todas as transações completam (*commit*), ou todas são desfeitas (*rolled back*)
    - Permite a atualização de bancos de dados em outros servidores
    - É transparente para os usuários

Two-Phase Commit

UPIS Graduação • Pós • MBA
CADA DIA MELHOR
CADA DIA MAIS UP

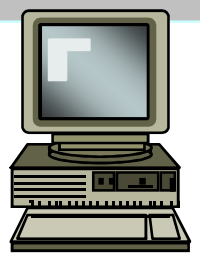
- 2PC – *Commit* em Duas Fases
  - Fases
    - ***Prepare***, um nó inicial (coordenador global) notifica os *sites* envolvidos na transação que se preparem
    - ***Commit***, se não houver nenhum problema todos os *sites* concluem suas transações (ou todos desfazem)

Implementação de Distribuição

```
update Pedido
set qt_prod = qt_prod + 10
where id_ordem = 4352
and id_prod = 56;

update Estoque
set qt_prod = qt_prod - 10
where id_prod = 56;

commit;
```



```
create synonym Estoque
for Estoque@ALMOXARIFADO;
```