分析型数据库 AnalyticDB

SQL开发指南

提纲

- DML
- SQL
- SQL 引擎说明
- SQL hint使用
- SQL 开发规范
- SQL开发客户端工具

DML

阿里云
aliyun.com

DML语句说明：

- 实时表（realtime）支持DML语句包括：INSERT和DELETE操作
- 批量表（batch）不支持INSERT和DELETE语句
- Update语句，当前版本需要通过Insert覆盖写方式实现update操作，2.8版本支持update
- 支持 Insert into tabname select … from tab 实现批量数据库内处理

DML – Insert 语句

阿里云 aliyun.com

语法：

```
INSERT INTO table_name [ ( column [, ... ] ) ] VALUES [(),()]
```

- 默认建表字段列顺序：INSERT INTO tabname values (?,?,?);
- 指定列名：INSERT INTO tabname(co1,col2,col3) values(?,?,?);
- 一次插入多条记录：INSERT INTO tabname(col1,col2,col3) values(?,?,?),(?,?,?),(?,?,?);
- 一次性提交16K为最佳实际性能，实际使用建议根据表行长来确定一次提交N条记录，N=16k/rowsize。
- 字符串类型支持单引号和双引号，如'abc'和"abc"

INSERT IGNORE vs INSERT

实时数据INSERT的默认行为为主键覆盖，即后INSERT的记录将覆盖系统中已有的相同主键的记录。 如果INSERT指定 IGNORE 关键字，若在系统中已经有相同主键的记录，则当前INSERT IGNORE语句执行能成功，但是新记录将会被丢弃掉。 用户可以根据业务应用的实际需求，选择INSERT IGNORE或者INSERT。

DML – INSERT FROM SELECT

阿里云 aliyun.com

语法示例：

```
INSERT INTO db_name.target_table_name (col1, col2, col3)
SELECT col1, col2, col3 FROM db_name.source_table_name WHERE col4 = 'xxx';

INSERT INTO db_name.target_table_name
SELECT col1, col2, col3, col4 FROM db_name.source_table_name WHERE col4 = 'xxx';
```

- 提供列名的方式，确保SELECT的列表与提供的INSERT目标表的列匹配（顺序、数据类型）
- 不提供列名的方式，确保SELECT的列表与提供的INSERT目标表的列匹配（顺序、数据类型）

多种引擎 及异步执行

```
/*+run_async=true, engine=mpp, mppNativeInsertFromSelect=true*/
INSERT INTO  insert_from_select_test SELECT * FROM lineitem;
```

异步化执行返回异步化任务的ID，ASYNC_TASK_ID，可通过该ID查询异步任务的状态：

```
SELECT * FROM information_schema.async_task WHERE id = '100_81_136_60_19010_1501141772740';
```

DML – Delete

阿里云 aliyun.com

语法：

```
DELETE FROM table_name [ WHERE condition ]
```

- 对于含有二级分区的表，Where condition 子句必须包含二级分区条件。
- delete from table_name where id=1 and biz_date>=20180201 and biz_date<=20180228

注意事项

- Delete操作在删除大量数据时，可能引发性能问题
- 特别是删除的记录跨越超过3个以上二级分区情况
- 如果是删除表所有记录情况，建议，通过drop table create table实现
- 如果表有大量历史数据需要清理，建议使用二级分区的自动清除机制来实现历史数据的自动删除
- 批量表的删除，需要通过修改表的二级分区数，或者全部从ODPS重新覆盖导入数据

DML – 批量表如何删除数据？

- ADB的批量表不支持insert/delete操作，那么如何实现批量表的数据删除?

操作方法：

1. 修改表的二级分区个数（比如从24修改到20）
2. 发起一次改表ODPS到ADB的任意一个同步任务
3. 查询表的数据，确认历史分区是否已经删除

备注：正常情况下批量表会自动根据二级分区数自动清除超二级分区数定义的历史分区

提纲

- DML
- SQL
- SQL 引擎说明
- SQL hint使用
- SQL 开发规范
- SQL开发客户端工具

SQL语法—Select – 兼容MySQL语法

```
SELECT
    [DISTINCT]
    select_expr [, select_expr ...]
    [FROM table_references
    [WHERE filter_condition]
    [GROUP BY {col_name | expr | position}
    [HAVING having_condition]
    [ORDER BY {col_name | expr | position}
        [ASC | DESC], ...]
    [UNION ALL (SELECT select_expr..)]
    [{UNION|INTERSECT|MINUS} (SELECT select_expr..)]
    [LIMIT {row_count}]
```

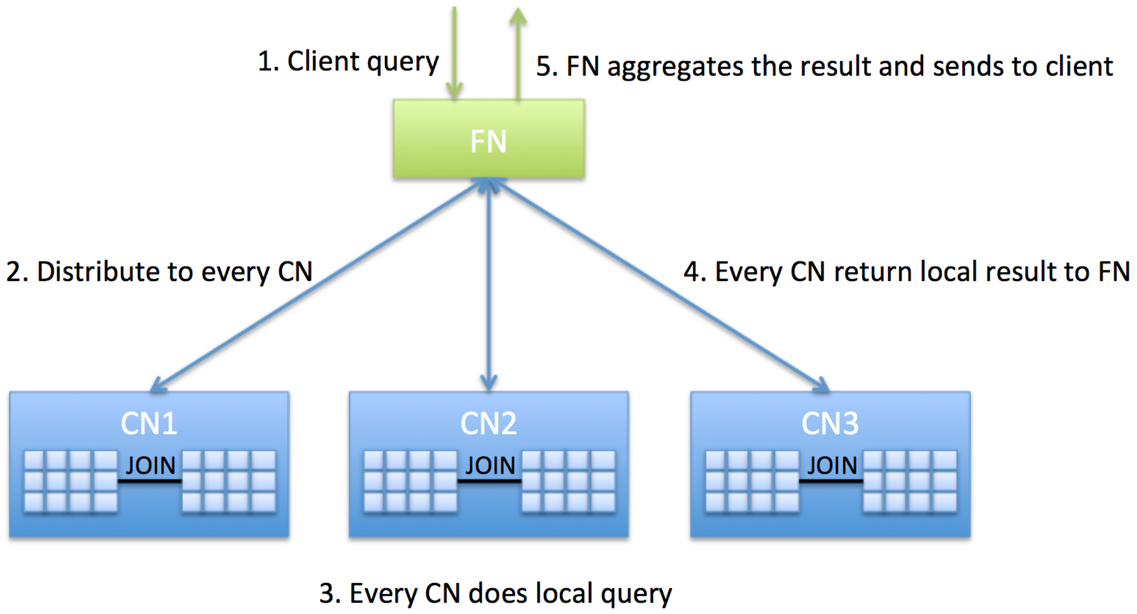
提纲

- DML
- SQL
- **SQL 引擎说明**
- SQL hint使用
- SQL 开发规范
- SQL开发客户端工具

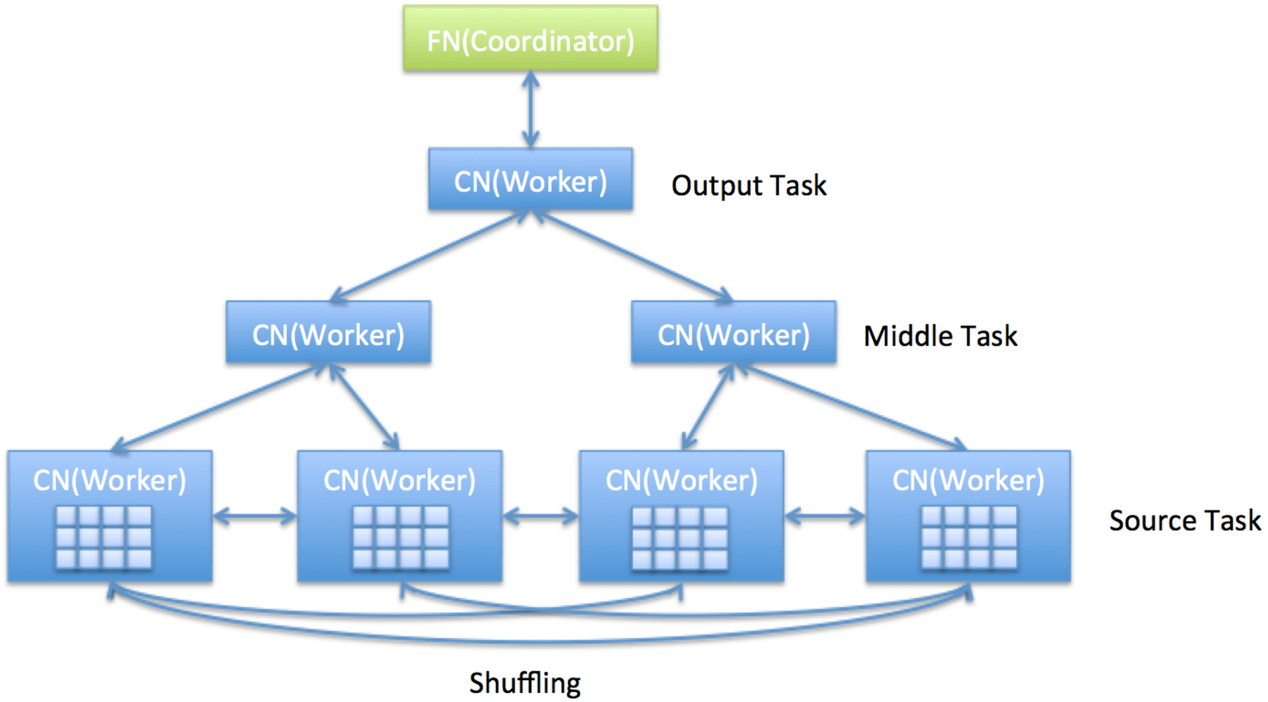
AnalyticDB 计算引擎

ADS目前支持两种计算引擎：**COMPUTENODE Local/Merge**(简称:Two-Stage)和**MPP**

Two-Stage：



MPP：



关于2种计算引擎的说明

COMPUTENODE Local/Merge(简称LM)：又称两阶段，优点是计算性能很好，并发能力强，缺点是对部分跨一级分区列的计算支持较差。
Full MPP Mode（简称MPP）：支持更丰富的函数，SQL语法，数据量计算能力。

默认计算引擎，V2.8 版本之后默认 MPP，之前版本默认LM。可以按Db切换默认引擎

引擎SQL hint：
/*+engine=mpp*/ select …. From …where …
/*+engine=*COMPUTENODE*/ select …. From …where …

提纲

阿里云
aliyun.com

- DML
- SQL
- SQL 引擎说明
- SQL hint使用
- SQL 开发规范
- SQL开发客户端工具

SQL Hint

1. 强制使用COMPUTENODE ：/*+engine=COMPUTENODE*/ SELECT ...
2. 强制使用Full MPP Mode计算引擎：/*+engine=MPP*/ SELECT ...
3. */*+no-index=[b.aa2]*/* SELECT a1, a2, COUNT(DISTINCT a3) rs FROM a JOIN b ON a.a1 = b.a1 AND b.a4 IN ('110', '120') AND a.aa3 = 1003 AND b.aa2 <= 201503 GROUP BY a1, a2;
4. */*+no-cache=[colname]*/* SELECT ...
5. **多个hint** /*+engine=COMPUTENODE,batchio=true,no-index=[nb_tab_mee.capture_time],blockcache=false,prefetch-step=4*/

提纲

- DML
- SQL
- SQL 引擎说明
- SQL hint使用
- SQL 开发规范
- SQL开发客户端工具

索引—智能全索引

ADS为表的所有列创建索引，（支持disable某些列的索引，默认创建索引）

SQL开发不需要考虑索引是否存在，但需要合理使用索引

SQL开发规范与示例—单表性能最佳SQL示例

阿里云 aliyun.com

简单的分库分表

```
select
 sum( case when t.op_1_timestamp >= 20171128000000 and t.op_1_timestamp
  and(( t.op_2_timestamp is null or t.op_2_timestamp > 20171128235900 o
 and( t.op_3_timestamp is null or t.op_3_timestamp > 20171128235900 or
and( t.op_4_timestamp is null or t.op_4_timestamp > 20171128235900 or t
from
    t_fact_mail_status t
where
    t.org_code = '21111101'
    and t.biz_date = 20171128
```

一级分区列
二级分区条件

SQL开发规范与示例--表关联性能最佳SQL示例

表join：保证：Local Join

- 一级分区键join
- 一级分区数一致

```
/* +engine=MPP */
SELECT
case when cus.age<=30 then '<20' when cus.age>20
and cus.age<=30 then '20-30' when cus.age>30
and cus.age<=40 then '30-40' else '>40' end as age_range,
COUNT(distinct cus.customer_id),
SUM(total_price)
FROM
t_fact_customers cus left join t_fact_orders ord
on  cus.customer_id=ord.customer_id
where ord.order_time>='2017-10-01 00:00:00' and ord.order_time<'2017-11-01 00:00:00'
AND ord.order_date=201710
group by age_range;
```

SQL开发规范与示例——一级分区裁剪

当要求高QPS查询业务时，需要从表的设计和SQL上利用分区裁剪能力。

```
select
    sum(mail_id) as cn
from
    t_fact_mail_status t
where
    t.org_code = '21111101'
    and t.biz_date = 20171128
```

SQL开发规范与示例—二级分区裁剪

包含二级分区情况，SQL中增加二级分区条件，减少二级分区扫描

```
1  select c1,c2 from tab1 where id = 3
2  and time between '2016-04-01 00:00:00' and '2016-04-01 12:00:00';
```

增加二级分区过滤条件，如果根据业务场景确认满足time between ‘2016-04-01 00:00:00’ and ‘2016-04-01 12:00:00’ 的二级分区列为20160401，则可以将该SQL改写为：

```
1  select c1,c2 from tab1 where id = 3
2  and time between '2016-04-01 00:00:00' and '2016-04-01 12:00:00'
3  and pid = 20160401;
```

多表关联--尽量的充分的过滤条件

多表关联查询，where条件中，需要显示的写明每一个表的过滤条件。通常我们习惯在传统数据库中，都是通过索引字段关联来快速检索数据。如下SQL:

```
Select count(*)
from t1 join t2
on t1.id=t2.id
where t1.time between '2017-12-10 00:00:00' and '2017-12-10 23:59:59'
and t1.type= 100
```

在明确t2与t1表都有同样的time和type过滤条件情况下， 建议修改为如下SQL:

```
Select count(*)
from t1 join t2
on t1.id=t2.id
where t1.time between '2017-12-10 00:00:00' and '2017-12-10 23:59:59'
and t1.type= 100
and t2.time between '2017-12-10 00:00:00' and '2017-12-10 23:59:59'
and t2.type=100
```

SQL常见问题

- **Select * 扫描全部列**

  select * from tab1 where c1>100 and c1<1000;

- **模式匹配**

  select c1,c2 from tab1 where c1=3 and c3 like '%abc%’;

- **函数导致索引无效**

  select c1,c2 from tab1 where substr(cast(time as varchar),1,10)='2017-10-12';

  select … from. .. WHERE date_format(base.biz_occur_date, '%Y%m%d')= date_format(date_add('minute',- 10, cast('2018-02-04 23:10:57' as timestamp)), '%Y%m%d')

- **去掉不必要的Is not null**

  Select c1,c2 from tab1 where c1>100 and c1<1000 and c1 is not null;

提纲

- DML
- SQL
- SQL 引擎说明
- SQL hint使用
- SQL 开发规范
- SQL开发客户端工具

SQL 开发客户端工具--DMS

阿里云
aliyun.com

DMS：

DMS是阿里云提供的Web开发管理工具，用户可以通过DMS进行数据库对象的管理操作及SQL开发：

- 创建／删除数据库
- 创建／删除表组
- 创建／删除表
- 权限管理
- 查看表DDL，修改表Alter table
- SQL开发

SQL 开发客户端工具—MySQL命令行客户端

阿里云 aliyun.com

MySQL命令行工具：

AnalyticDB支持通过MySQL客户端直接连接AnalyticDB数据库，连接命令行如下：

mysql -hhost -Pportnumber -Ddbname -uusername -ppassword -A -c

- host：为AnalyticDB对外提供的hostname，找管理员通过dms管理控制台获取
- portnumber：为AnalyticDB对外提供的端口号，找管理员通过dms管理控制台获取
- dbname：为AnalyticDB对外提供的数据库名，找管理员通过dms管理控制台获取
- username, password：为用户的AK，通过阿里云管理平台获取
- 选项-c，支持AnalyticDB SQL Hint

使用示例：

执行多个SQL语句文件, -v选项输出SQL语句

mysql -hhostname -P9999 -Dtestdb -uusrnameak -ppassword -A -c -v < 1.sql

后台执行SQL任务

mysql -hhostname -P9999 -Dtestdb -uusrnameak -ppassword -A -c -v < test.sql > test.log 2>&1 & done

SQL 开发客户端工具—第三方图形化工具

阿里云 aliyun.com

支持MySQL客户端：DBeaver

DBeaver是一个非常专业的、通用的数据库管理工具和 SQL 客户端，是目前支持数据库版本最广泛的SQL开发工具之一

- DBeaver 提供一个图形界面用来查看数据库结构，表DDL语句
- 执行SQL查询和脚本
- SQL 格式化
- 自动生成SQL
- 浏览和导出数据
- DBeaver和AnalyticDB进行了深度适配，可以直接通过MYSQL jdbdc driver连接AnalyticDB数据库。

下载地址：

https://dbeaver.jkiss.org/download/

Thanks!

咨询邮箱：ADB_SUPPORT@service.alibaba.com